Design Principles for Lifelong Learning AI Accelerators

Dhireesha Kudithipudi[1*], Anurag Daram[1], Abdullah M. Zyarah[1], Fatima Tuz Zohora[1], James B. Aimone[2], Angel Yanguas-Gil[3], Nicholas Soures[1,4], Emre Neftci[5], Matthew Mattina[6], Vincenzo Lomonaco[7], Clare D. Thiem[8], Benjamin Epstein[9]

[1*]University of Texas at San Antonio, San Antonio, TX, USA.
[2]Sandia National Laboratories, Albuquerque, NM, USA.
[3]Argonne National Laboratory, Lemont, IL, USA.
[4]Rochester Institute of Technology, Rochester, NY, USA.
[5]Forschungszentrum Jülich and RWTH Aachen, Aachen, Germany.
[6]Tenstorrent Inc., MA, USA.
[7]University of Pisa, Pisa PI, Italy.
[8]Air Force Research Laboratory, Rome, NY, USA.
[9]ECS Federal, Arlington, VA, USA.

*Corresponding author(s). E-mail(s): dk@utsa.edu;

Abstract

Lifelong learning — an agent's ability to learn throughout its lifetime — is a hallmark of biological learning systems and a central challenge for artificial intelligence (AI). The development of lifelong learning algorithms could lead to a range of novel AI applications, but this will also require the development of appropriate hardware accelerators, particularly if the models are to be deployed on edge platforms, which have strict size, weight, and power constraints. Here, we explore the design of lifelong learning AI accelerators that are intended for deployment in untethered environments. We identify key desirable capabilities for lifelong learning accelerators and highlight metrics to evaluate such accelerators. We then discuss current edge AI accelerators and explore the future design of lifelong learning accelerators, considering the role that different emerging technologies could play.

Lifelong learning (also known as continual learning) is an approach to artificial intelligence (AI) in which a model is expected to learn from noisy, unpredictable, and changing data distributions while continually consolidating knowledge about new information. The model must transfer previously acquired knowledge forward to new tasks, transfer new knowledge backward to previously learnt tasks, and adapt quickly to contextual changes [1, 2]. The capabilities of AI systems have advanced notably in recent years, but designing lifelong learning machines remains difficult. Algorithmic-level innovations are important to address this problem. However, to be of practical value, lifelong learning models must often be deployed on physical hardware at the edge, under strict size, weight, and power constraints. And this will require advances in hardware accelerators for lifelong learning.

Current AI accelerators with on-device learning capabilities support aspects of lifelong learning. (Note, lifelong learning and continual learning are often used interchangeably, but continual learning is not equivalent to online on-device learning.) Edge AI accelerators, in particular, contain limited computational capabilities and operate under battery power, and various challenges — including those related to dataflow, external memory access and computation — remain to be addressed with these systems. The workload profile for lifelong learning also has different characteristics on the edge. These characteristics include being able to process data at variable frequencies while learning relevant features from them, operating under memory and computational constraints, and optimising for energy-accuracy trade-offs in real-time. The characteristics also limit the direct application of optimisation techniques often used in cloud environments. In addition, few of the design approaches provided for edge AI accelerators can transfer to large-scale systems.

In this Perspective, we examine the development of lifelong-learning capable digital hardware accelerators for untethered devices. We first consider fundamental aspects of lifelong learning algorithms and then identify the hardware design requirements that digital accelerators must meet in order to support lifelong learning. We discuss the importance of standardised metrics for lifelong learning systems, and provide an overview of current accelerator designs. We also explore the future of lifelong learning accelerator designs and consider the role that different emerging technologies could play.

Fundamentals of lifelong learning

The term *lifelong learning* refers to a system's ability to autonomously operate in, interact with, and learn from its environment [3–5]. This requires the system to be able to improve its performance through the acquisition of new knowledge and learning to operate under energy and resource constraints. *More specifically, a lifelong learning system needs to function in dynamic and noisy environments, rapidly adapt to novel situations, minimise forgetting when learning new tasks, transfer information between tasks, and operate without explicit task identification.*

Several learning paradigms and features have been involved in the process of arriving at this definition of lifelong learning. The concept includes transfer learning in

which the goal was to recycle/reuse the learnt representations for other tasks [6]. This was followed by multi-task learning (MTL), which had the goal of improving generalisation by leveraging domain-specific information in related tasks [7]; however, in this setting, tasks are trained jointly and not sequentially. Transfer learning and MTL evolved into few-shot learning [8], in which the goal is to learn from a limited number of examples with supervised information in the target domain. This drove the approach towards learning to learn (also called meta-learning), where a system learns to optimise the objective on its own; an approach that aims to achieve the rapid, general adaptation that biological brains can offer [9]. In tandem, studies on biological brains have shown that there is a combination of learning mechanisms that support lifelong learning, rather than a single one [1]. Drawing from this and more, researchers consider that a number of these learning paradigms constitute a subset of lifelong learning .

Methods have been devised to address different aspects of lifelong learning. These methods are *synaptic consolidation*, *dynamic architectures*, and *replay* (Fig. 1).

Synaptic consolidation is a way to preserve synaptic parameters when learning new tasks. The most common method is to add regularisation terms to the loss function to maintain previous synaptic strengths or neural activations [10, 11]. Another approach is to use more complex synapse models with memory-preserving mechanisms such as metaplasticity [12], multiple weight components operating on different timescales [13], or probabilistic synapses [14].

Dynamic architectures expand network capacity in order to solve a particular task, using top-down control of resources. The expansion of network capacity takes different forms, including periodic additions of new neurons or addition of entire new networks or layers dedicated to solving new tasks [15, 16]. Another approach is to dynamically gate the network components according to the task identity. This is often achieved by means of a task oracle (that is, an autoencoder capable of separating specific classes and tasks) or by meta-learning a gating function [17].

Replay involves the presentation of samples representative of previously learnt tasks interleaved with samples from the task currently being trained. Replay helps bring the training data closer to being independent and identically distributed, as would be the case if all tasks had been trained jointly. This allows the network to be trained to maximise performance across all tasks (as opposed to only learning the current task) and to learn inter-task boundaries. To replay data in neural networks, previous training samples (or internal representations) are encoded, stored, and recalled from a memory buffer or recreated by a generative model [18–20]. Methods are evolving to use more sophisticated algorithms for both the selection of samples to be stored in the replay buffer and for the selection of samples to be replayed from the buffer [21].

Several machine learning models have been developed to address lifelong learning, with a heavy emphasis on the catastrophic forgetting problem using the methods discussed above, as well as some hybrid models that combine features from two or more of them. Increasingly, AI researchers are taking inspiration from discoveries in neuroscience that help explain how biological organisms are able to learn continually [1].

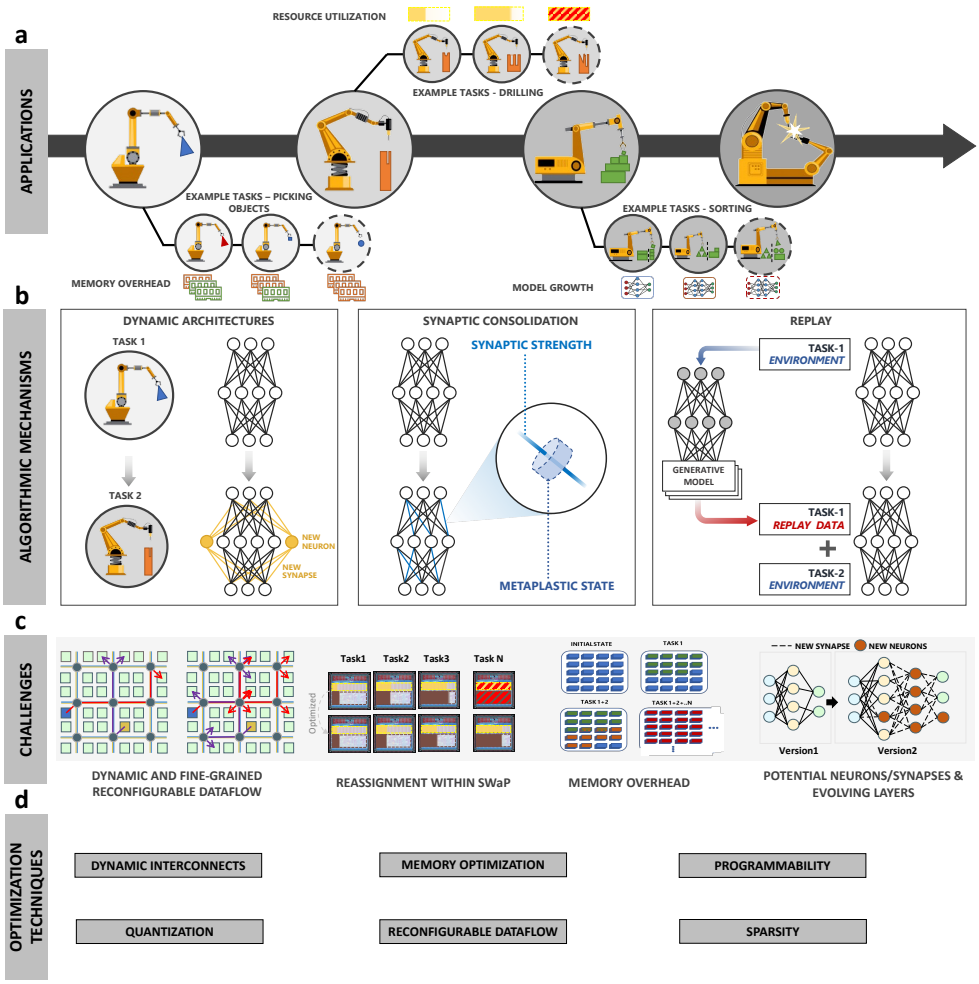

Fig. 1 Addressing lifelong learning in AI systems - **a**, Applications : Lifelong learning shown in the context of sequential tasks (large circles) and sub-tasks(smaller circles) with varying degrees of similarity, and the associated hardware challenges. **b**, Algorithmic mechanisms: A broad class of mechanisms that address lifelong learning. Dynamic architectures either add or prune network resources to adapt to the changing environment. Regularization methods restrict the plasticity of synapses to preserve knowledge from the past. Replay methods interleave rehearsal of previous knowledge while learning new tasks. **c**, Hardware challenges: Lifelong learning imposes new constraints on AI accelerators, such as the ability to reconfigure datapaths at a fine granularity in real-time, dynamically reassign compute and memory resources within a size, weight, and power(SWaP) budget, limit memory overhead for replay buffers, and generate potential synapses, new neurons and layers rapidly. **d**, Optimization techniques (Bottom): Hardware design challenges can be addressed by performing aggressive optimizations across the design stack. Few examples are dynamic interconnects that are reliable and scalable, quantization to >4-bit precisions during training, hardware programmability, incorporating high bandwidth memory, supporting reconfigurable dataflow and sparsity.

Key capabilities for lifelong learning accelerators

Recent research on synaptic consolidation [10, 12–14], dynamic architectures [15, 16] and replay methods [18, 20] show promise in addressing aspects of lifelong learning. By studying these methods, we have assembled a list of six desirable capabilities for AI accelerators: on-device learning, resource reassignment within budget, model recoverability, synaptic consolidation, structural plasticity, and replay memory.

The first three capabilities are general ones that apply to any lifelong learning method; the last three are specific to one or more of the lifelong learning methods discussed above. The relevance of each capability to a specific design will depend on the type of accelerator and the target application.

On-device learning

A lifelong learning system needs to continuously learn from non-stationary data distributions over varying time-scales [22]. Since the system has to learn in real-time, on-device learning is crucial to avoid latency associated with data transmission to the cloud. Generally, for on-device learning, design considerations are required i) when batching data of large samples on limited memory resources and ii) when storing and mapping a large pool of intermediate model parameters in compact formats to minimise data movement cost. While translating to the context of lifelong learning, additional challenges arise: optimising complex computations observed in consolidation methods so that the energy cost is reduced and minimising the latency to access previous samples in replay methods.

Resource reassignment within budget

Lifelong learning systems must have the ability to reallocate or distribute resources within a size, weight, area, and power (SWaP) budget at runtime, and quite often this could entail different granularities. This suggests that the system must be capable of parametric, neuronal or memory reassignments during its lifetime [23]. This is a challenging requirement — especially when hardware is tightly optimised for one method, model, or functionality. Furthermore, to allocate resources at fine-granularity one needs to identify points of operation that are pareto-optimal, while ensuring that the architecture remains flexible to different tasks. Additional challenges may arise when the size of the input and output layer differs between tasks [24, 25], such as accommodating the model expansion on the fly and/or the inability to encode and store data within a fixed size.

Model recoverability

One challenge in developing a lifelong learning AI system is to establish confidence in the model, and more so when the system is changing autonomously [26]. In a software environment, it is possible to checkpoint a model's state, preserving the overall model or maintaining a history of updates. This tracking provides a valuable record of a model's past states that can prove essential for either diagnostic purposes or for reverting to a previous state (if an online update led to failure). Several of the

design choices that make AI accelerators more efficient make checkpointing a model's configuration dynamically impractical, if not impossible.

Synaptic consolidation

Models incorporating synaptic consolidation typically use multiple internal states to learn at different timescales using several loss functions, probabilistic synapses, reference or target weight values or other synaptic states in addition to magnitude (*i.e.*, metaplastic state, consolidation tag). Each of these consolidation mechanisms entails auxiliary information stored on-device and additional operations performed during any learning process. In general, for consolidation mechanisms, a lifelong learning accelerator needs to store and associate auxiliary information with specific synapses/neurons and support custom loss functions.

Structural plasticity

Structural plasticity relates to physical changes in the model, including the addition/removal of synapses (e.g. synaptogenesis or synaptic pruning) or of neurons (e.g. neurogenesis/neural pruning), gating and attention mechanisms, and mixture of experts [16, 27, 28]. While static allocation of pools of neurons and/or synapses is possible at design time, it is still challenging to model and train such highly dynamic architectures. The underlying accelerators should support fine-grain runtime reconfigurability to add or reallocate a new pool of memory and computational resources. This problem is exacerbated when reallocation has to occur under a limited SWaP budget.

Replay memory

Replay mechanisms might require a continuously growing memory as the model learns new tasks. Replay comes in two varieties, known as waking and sleeping replay, both of which need to be supported by accelerators. In waking replay, to prevent forgetting previously learnt tasks, selected samples representative of the earlier tasks are interleaved while training a new task. During sleep replay, the system rehearses only samples from previous tasks to consolidate knowledge. Memory access during sleep replay can be at a slower clock cycle, since the model does not need to respond to real-time stimuli. Memory storage can be off-chip. Waking replay, on the other hand, requires on-chip buffers that can update the system state at a faster rate without disrupting learning on the current task. Overall, memory storage and access patterns for replay mechanisms are of mixed latency and are distributed.

The current generation of accelerators includes some highly programmable devices, but there are not any that support the full set of capabilities listed above. And those that do support some of the features [29, 30] do not meet the size, weight, area, and power (SWaP) budget constraints imposed by untethered devices.

Initial metrics to evaluate lifelong learning accelerators

In lifelong learning scenarios, the statistical properties of the input stream cannot be assumed to be stationary, an assumption that underlies traditional statistical learning theory approaches. This limits the usability of standard evaluation protocols and metrics described in the machine learning literature. However, the methods used to assess the quality and capability of lifelong learning systems have been evolving along with the progress in models and applications [31].

Table 1 lists recent lifelong learning metrics for evaluating algorithms. Based on pilot implementations [32, 33], we also propose additional metrics for lifelong learning accelerators (communication overhead and multi-tenancy).

A common benchmark for assessing how a system learns a sequence of tasks is to measure the mean accuracy across all tasks after the model has been trained on each task at least once (that is, at the end of the overall training sequence). This metric can be compared to the accuracy achieved when all tasks are trained jointly to obtain a measure of the amount of forgetting that is due to sequential learning. However, there are nuances that are not reflected in this type of measurement. For example, it does not distinguish between a system that learns the first task perfectly but is unable to learn subsequent ones and a system that learns the last task perfectly but forgets the preceding ones. For this reason, studies [19, 34] that measure how much performance changes in prior tasks (backward transfer), and how performance changes in downstream tasks (forward transfer) have provided more insight into how systems address continual learning and where their limitations lie.

Other metrics specific to lifelong learning include more granular methods for measuring trade-offs between plasticity and stability, measuring how continual learning can be leveraged to learn faster or improve performance compared to a baseline model (by using prior knowledge), measuring the time to recover performance after task transitions (performance recovery), and measuring performance degradation on previous tasks after each new task is learnt (performance maintenance). Beyond performance, it is also important to study models in terms of applicability [35]; this includes metrics quantifying how robust models are to noise, failures and data ordering, and autonomy of the model (for example, does it need supervision, task oracles).

Another important dimension of evaluation, especially for edge AI accelerators, is sample efficiency and scalability. These aspects can be quantified in terms of memory overhead, training speed, and network growth over time (which should preferably be bounded irrespective of the amount of data processed). Furthermore, for the accelerators, we consider the cost of data movement and reuse for all the methods (arithmetic intensity), energy efficiency of the system, total memory footprint that can increase with lifelong learning methods, the communication overhead that is associated with accessing data from higher-order memory for replay or plasticity, and multi-tenancy capturing the systems response rate when executing sequential tasks for real-time operations.

In addition, there are metrics and evaluation protocols specific to accelerator design. For example, metrics that highlight the efficiency-efficacy trade-off, method tunability as it depends on specific application requirements [36], performance variability as a function of data sequence length and out-of-distribution streams, and continual evaluation regimes that measure worst-case performance, necessary when

Table 1 Overview of current metrics for lifelong learning algorithms [31] and proposed metrics for the accelerators.

Current Metric	Formula	Metric Assessment of System
Mean Accuracy (MA)[1]	$\text{MA} = \sum_{t=1}^{N} \frac{\text{R}^{t,N}}{N}$	Average performance of model on all the tasks experienced
Memory Overhead (MO)[2]	$\text{MO} = min\left(1, \frac{1}{N} \sum_{i=1}^{N} \frac{Mem(\theta_i)}{Mem(\theta_b)}\right)$	Average overhead in memory a model requires per task
Forward Transfer (FWT)	$FWT = \frac{1}{N-1} \sum_{k=1}^{N-1} \sum_{t=k+1}^{N} \frac{\text{R}^{t,k} - R^{t,k-1}}{N-k}$	Average change in accuracy across all tasks $t > k$, after learning task T^k
Backwards Transfer (BWT)	$BWT = \frac{2}{N(N-1)} \sum_{k=2}^{N} \sum_{t=1}^{k-1} \text{R}^{t,k} - \text{R}^{t,k-1}$	Average change in accuracy on tasks $t < k$, after learning task T^k
Performance Recovery (PR)[3]	$PR = \frac{d}{dt}(T_{recovery}(t))$	Slope of recovery times of model in response to a change
Performance Maintenance (PM)	$PM = \frac{1}{N-1} \sum_{k=1}^{N-1} \sum_{t=k+1}^{N} \frac{(\text{R}^{k,t} - \text{R}^{k,k})}{N-k}$	Average change in performance on a task after learning new tasks
Sample Efficiency (SE)[4]	$SE = \frac{\text{R}^{t,t}}{b^t} \times \frac{T_{sat}^{b^t}}{T_{sat}^{\text{R}^{t,t}}}$	Efficiency of a lifelong learner vs a single task expert to reach saturation in performance
Arithmetic Intensity[5]	$\frac{OPs}{Byte}$	Reuse efficiency of accelerator as number of operations per byte of memory traffic
Energy Efficiency	$\frac{OPs}{W}$	Efficiency of the system as a ratio of computing throughput of the system per W
Learning Cost	$N_{trainsteps} \times N_{updates} \times \frac{Energy\text{-}Cost}{Update}$	Energy cost for the training process of the accelerator
Area Efficiency	$\frac{OPs}{mm^2}$	The number of operations per each mm^2 of the chip for a given technology node
Working Memory Footprint	$Bytes$	Net memory size required for learning different tasks
Communication Overhead,*[6]	$f(D, C)$	Cost of communication as a function of the distance between two memory accesses (D) and associated memory access cost (C)
Multi-Tenancy,*[7]	$\Delta(T_{(t+1)}, T_t + L_t)$	Time lapse between the runtime of two sequential tasks

[1] $\text{R}^{t,k}$ represents the accuracy of the lifelong learning on task number t after learning task number k. N represents the total number of tasks the model experiences with k, $t \leq N$

[2] θ_i represents the average amount of memory a model requires per task, and θ_b baseline model's memory size.

[3] $T_{recovery}(t)$ is the function of the curve formed by the recovery times for the lifelong learning model. Recovery time measures the time to recover its performance when a change is observed.

[4] b^t represents the performance of a single task expert model on a task t. $T_{sat}^{b^t}$ and $T_{sat}^{\text{R}^{t,t}}$ denote time to performance saturation (T can be represented by number of samples to reach saturation) for single task expert and lifelong learner, respectively. SE measures the ratio of time (number of samples) to reach saturation scaled up by the ratio of peak performance for a single task expert and the lifelong learning model.

[5] OPs refers to the number of compute operations required for performing a task and *Byte* refers to the number of bytes accessed in memory. This provides the ratio of number of operations per byte of memory traffic.

[6] D refers to the distance of the data in memory hierarchy and C refers to the cost of memory access.

[7] Latency of the task L = $\frac{instructions}{task} \times \frac{cycles}{instruction} \times \frac{seconds}{cycle}$, T_t is the time at which task t starts.

* Represent the newly proposed hardware metrics, which are important for evaluating lifelong learning accelerators.

deploying critical real-world applications [37]. Several metrics and benchmarking platforms have been published for AI inference accelerators [38, 39] and lifelong learning algorithms [31], but there is a need to identify and develop new benchmarks and workloads suitable for continual learning accelerators on a larger scale, including hardware and algorithmic metrics.

Lifelong learning on current untethered AI accelerators

AI accelerators can be categorised based on those that support traditional rate-based implementations and those that support spiking neural networks. Tables 2 and 3 provide an overview of digital AI chips (rate-based and spiking) that can perform on-device learning in untethered environments. We split the accelerators into two tables [40] because the baseline resolution of the computations and associated metrics such as performance (TOP/s) and power are different for spiking accelerators and rate-based accelerators and because SNN algorithms generally support different network topologies and have different encoding schemes. We also note there are differences in the choice of design optimisations for the two sets of accelerators.

Traditional accelerators for rate-based AI algorithms have primarily been focused on implementing and optimising DNNs, CNNs, and RNNs, generally trained using gradient descent applied with backpropagation. Design choices during the development of these accelerators focus on microarchitectural exploration [41, 42], energy-efficient memory hierarchies [49, 50, 51], flexible dataflow distribution [43–45], domain-specific compute optimisations such as quantisation, pruning, and compression [46–48], and hardware-software co-design techniques [49]. More recently, a rate-based accelerator with latent replay and on-device continual learning has been proposed [36]. The design leverages the benefits of quantisation and tiling to reduce compute and memory overheads.

Compared to rate-based accelerators, spiking neuromorphic accelerators are designed to support algorithmic models that more closely mimic the functionality of their biological counterparts. This often involves more complex synaptic and neuronal dynamics, which are inherently temporal in nature. The typical characteristic of a spiking neuron is how it integrates information over time, and only releases a spike once the accumulated information crosses a threshold. Neuromorphic systems tailored for SNNs have demonstrated efficiency in a number of ways: low cost of a single spike, asynchronous and sparse communication, and cheaper synaptic operations that do not require multiplication. Though these features can be achieved in non-spiking domains, unlike recurrent neural networks, SNNs have an inherent temporal aspect in the neuron dynamics without the need for recurrent connections while offering greater applicability and computational power than binary neural networks.

In the context of lifelong learning, several promising methods draw inspiration from neural plasticity, including spike-timing-dependent plasticity, heterosynaptic weight decay and consolidation, and neuromodulation [1]. The bio-plausible learning models generally have local learning, neuronal and synaptic plasticity rules that require fine granular control on the hardware substrates. Neuromorphic accelerators inherently

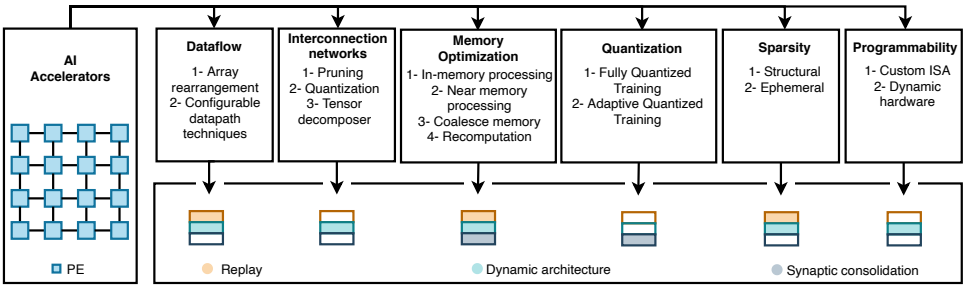

Fig. 2 Hardware optimisations for lifelong learning in AI accelerators. The bar plots indicate which lifelong learning features are affected by each optimisation technique.

offer a high degree of freedom to optimise dynamically at such fine granularity compared to the rate-based accelerators [50]. For example, triplet STDP rules [51] that learn temporal correlations between pre- and post-synaptic spike neurons (useful for rapid adaptation) are more amenable for deployment on neuromorphic accelerators that support integrate-fire dynamics.

The current spiking accelerators capable of learning can be divided into two broad categories: large-scale spiking accelerators[50, 52, 53] and accelerators targeted towards edge platforms [54–59]. Large-scale spiking accelerators support a wider range of applications or cortical simulations that focus on scalability and programmability [50, 52, 53]. These accelerators employ multiple independent parallel cores to realise the neurons and synapses of the network. The cores are connected through network-on-chip interconnects, which offer greater flexibility with high-bandwidth inter-chip or inter-board interfaces. In contrast, accelerators targeted towards edge platforms consist of a single core that supports various degrees of network connectivity, such as full connectivity in a crossbar architecture or locally-competitive- algorithm based architectures for sparse inputs [60].

Although most accelerators in both categories usually incorporate local unsupervised learning for on-device training, recent efforts are advancing toward incorporating multilayer spiking neural networks with supervised training [55]. There have also been extensions of supervised SNN models to lifelong learning. For example, a digital spiking network was designed for lifelong learning tasks using surrogate gradient-based training [33]. This accelerator uses activity-dependent metaplasticity to enable lifelong learning. For example, memory access overhead is reduced by using sparse spike-based communication, where only indices of neurons with active spikes are transmitted. Quantisation and compute-lite linear metaplasticity functions reduce memory and computational overhead.

Optimisation techniques

There are a number of optimisation techniques used in current AI accelerators: reconfigurable dataflow, memory optimisation, dynamic interconnection networks, quantisation, sparsity, and programmability. For each of these techniques, we suggest extensions that would facilitate lifelong learning (Fig. 2). We also highlight how each method may impact the metrics described in Table 1. However, it is important to note

Table 2 Overview of AI chips with on-device training and feature optimisations that can support a subset of the features of lifelong learning in untethered environments.

Chip	Quantization	Neural Network(s)	Power (W)	Performance[1][7]	On-chip Memory[2][6]	Energy Efficiency[3][7]	Sparsity	Dataflow
PuDianNao [91]	FP16/FP32[1]	7 ML algorithms[2]	596 mW (65nm)	1.056 TOP/s	32 kB	1.77 TOPS/W	-	-
PNPU [78]	FP8, FP16	DNN, CNN	425 mW@1.1V (65nm)	0.31 TFLOP/s (FP16) 0.61 TFLOP/s (FP8)	338 kB	0.72 TFLOPS/W (FP16) 1.44 TFLOPS/W (FP8)	In/W/Out zero skipping[3]	-
GANPU [72]	FP8, FP16	GAN	647 mW@1.1V (65nm)	0.54 TFLOPS (FP16) 1.08 TFLOPS (FP8)	676 kB	0.83 TFLOPS/W (FP16) 1.66 TFLOPS/W (FP8)	In/Out zero skipping	Reconfigurable 2D mesh connection
Evolver [46]	INT2, INT4, INT8	DNN, CNN	36 mW@1.1V (28nm)	2.195 TOP/s (INT2×2) 0.137 TOP/s (INT8×8)	416 kB	173 TOPS/W (INT2) 32.9 TOPS/W (INT8)	In/Out zero skipping	Tile-wise dataflow reconfiguration
HNPU [47]	SDFXP 4/8/12/16[4]	DNN, CNN	1162 mW (28nm)	3.07 TOP/s (SDFXP4)	552 kB	50.3 TOPS/W (FXP4)	In-/Out-slice zero skipping	-
LNPU [44]	FGMP FP8-FP16[5]	DNN, CNN, RNN	367 mW (65nm)	>0.6 TOP/s (FP8)	372 kB	1.74 TOPS/W (16b) 3.48 TOPS/W (8b)	In-zero skipping	Reversible datapath Tiled weight rearrangement
DF-LNPU [45]	FXP13/16, FP8	MLP, CNN, RNN	424 mW@1.1V (65nm)	LC:0.151 TOP/S[7] ZCC:0.3-1 TOP/s[7]	337 kB	LC:0.61-1.1 TOPS/W[7] ZCC:1.7 TOPS/W[7]	In/W zero-skipping	Transpose in custom SRAM
Agrawal et al. [79]	Hybrid-FP8, FP16	MLP, CNN, RNN	n/a (7nm)	16 TFLOPS		0.98 TOPS/W	-	Flexible Datapath MUXs
SOVC18 [64]	FP16	MLP, CNN, LSTM	n/a (14nm)	1.5 TFLOP/s	2 MB	-	-	2D Torus
SSCL20 [62]	FXP16	CNN	299 mW@1V (65nm)	0.15 TOP/s	1.12 MB	0.5 TOPS/W	-	Diagonal storage pattern with bit rotator
ISSCC19 [48]	bfloat16, FXP16/8/4	MLP, CNN, RNN	196 mW@1.1V (65nm)	0.204 TOP/s	139 MB/20000 experiences[6]	2.16 TFLOPS/W (16b)	-	Transposable PE array datapath
CHIMERA [122]	INT 8	DNN, CNN	418 mW (40nm)	0.92 TOP/s	2 MB[6]	2.2 TOPS/W	Gradient sparsity	Weight Stationary
Tenstorrent Wormhole [123, 124]	FP16, FP8, bfloat16	DNN, CNN, LSTM	80 W (12nm)	430 TOP/S	120 MB	-	Activation and parameter sparsity	-
SIGMA [65]	FP16/32	DNN, RNN , CNN	22.3 W (28nm)	10.8 TFLOPS	68 MB	0.48 TFLOPS/W	Bitmap compression	Reduction tree microarchitecture

[1] Multiplication in FP16, accumulation uses FP32

[2] K-means, k-nearest neighbours, naive bayes, support vector machine, linear regression, classification tree and deep neural network.

[3] In/W/Out refer to Input/Weight/Output

[4] SDFXP - Stochastic dynamic fixed point representation

[5] FGMP - Fine-grained mixed precision

[6] All AI chips are using on-chip SRAM except [122] which uses RRAM. In case of [48], it is not reported.

[7] Energy efficiency and performance with no sparsity. In [45] ZCC refers to zero-skip convolution cores and LC is the learning core.

Table 3 Overview of spiking neural network chips with on-device training and feature optimizations that can support a subset of the features of lifelong learning in untethered environments.

Chip	Quantisation	Neural Network(s)	Power	Throughput	On-chip Memory [8]	Connectivity	Sparsity
Loihi [50, 125]	INT1-INT9 [1]	Spiking MLP, Spiking CNN, LSM	420 mW (14 nm)	50 FPS (10kHz) [2]	33 MB	NoC	Sparse activity-dependent asynchronous flow control
SpiNNaker 2 [73, 126]	FXP32, FP32	DNN, SNN CNN	$\sim$ 0.72 W (22nm) [3]	4.6 TOP/s (250MHz)	18MB SRAM [4]	NoC	DVFS based on input sparsity
BrainChip Akida [53]	INT1, INT2 INT4	CNN, SNN	434 mW (80 FPS, 28nm)	1.5 TOP/s (300MHz)	8 MB	NoC	Sparse event-driven computations
ODIN [56]	INT4	SNN (SDSP) [5]	477 μW (28nm)	37.5 MSOP/s (75MHz) [6]	36 kB (SRAM)	AER bus	Sparse event-driven computation
Intel SNN Chip [57]	INT7	SNN (STDP)	6.2 mW (inference, 10nm)	25.2 GSOP/s [6]	896 kB	AER NoC	Input/weight-based skipping
ReckON [55]	INT8	Spiking RNN	114-150 μW(28nm)	-	138 kB (SRAM)	AER bus	ET/STE-based skipping[7]
DANNA [58]	INT8	SNN (STDP)	n/a (130nm)	–	—	Nearest neighbor	—
SCOLAR [33]	FxP16	SNN	21.25 mW(65nm)	250 MOP/s (10MHz)	100 kB (SRAM)	NoC	Sparse event-driven computation

[1] Signed or unsigned integer or mixed-precision number is supported.

[2] FPS - Frames per second.

[3] Processing Element (PE) power.

[4] Spinnaker 2 is also equipped with 8GB off-chip DRAM.

[5] SDSP - Spike-driven Synaptic Plasticity.

[6] SOP - Synaptic Operations, MSOP - Mega Synaptic Operations, GSOP - Giga Synaptic Operations.

[7] ET- Eligibility trace, STE - Straight-through estimator of spiking activation function.

[8] All chips use SRAM as on-chip memory.

that the optimisation techniques listed can help to realise only a subset of features for lifelong learning.

Reconfigurable dataflow

On-device training requires iterative processing to compute optimal model parameters. The training procedure typically consists of three stages, forward pass, backward pass and parameter update. The three stages share a processing element (PE) array but have different dataflows, which leads to different memory access patterns for the same data array. In addition, the optimal memory layout is different for each data flow, making optimising the operations of all three training stages difficult. This incongruity between dataflow and memory layout causes redundant memory access operations (each memory access consumes $\sim$200$\times$ more energy than compute units [61]) and reduces core utilisation, resulting in speed and *energy efficiency* degradation. Recent studies have described several techniques to address this problem, which can be broadly classified into array rearrangement and configurable dataflow techniques. Array rearrangement techniques by performing matrix transpose in custom SRAM, introducing diagonal storage pattern [62] or tiled weight rearrangement [44, 63] are some examples that are currently used, and are presented in Table 2. Configurable dataflows [41, 64] include reversing dataflow for weights and outputs [44], or exchanging paths between weights and inputs [48].

When considering lifelong learning accelerators, novel dataflow optimisation techniques can play a large role in handling structural plasticity and performing efficient replay. Unlike the three stages observed in on-device learning using backpropagation, structural plasticity involves a dynamic change in network architecture, requiring different optimal memory layouts for each change. Similarly, for replay, the system transitions from learning from streaming data to accessing, processing, and learning from batched samples of previous experiences. Such dynamic behaviour significantly increases the search space for finding the mapping (translating the model into its hardware-compatible representation) that best optimises data movement for improved arithmetic intensity. Additionally, to accommodate this flexibility in structure, some studies have explored techniques such as reduction tree microachitectures, atomic dataflow using graph-level scheduling and mapping [65, 66] to handle sparse and irregular data movement, but trade speed for higher power consumption. The aforementioned techniques can be useful in supporting mechanisms such as episodic replay, pruning and dynamic gating; however, further complex architectural changes like neurogenesis or the addition of entirely new layers or networks in real-time pose challenges in adopting high degrees of reconfigurability and identification of optimal mapping space without impacting *energy efficiency* of the accelerator.

Memory optimisation

Off-chip memory access can account for more than 80% of total energy consumption at inference [67]. Consequently, during training the overhead is more pronounced, since calculating gradients can require up to 10$\times$ as much memory access as updating the weights [41]. Therefore, it is crucial to reduce energy consumption due to memory access for energy-constrained platforms. One way to achieve this goal is to devise

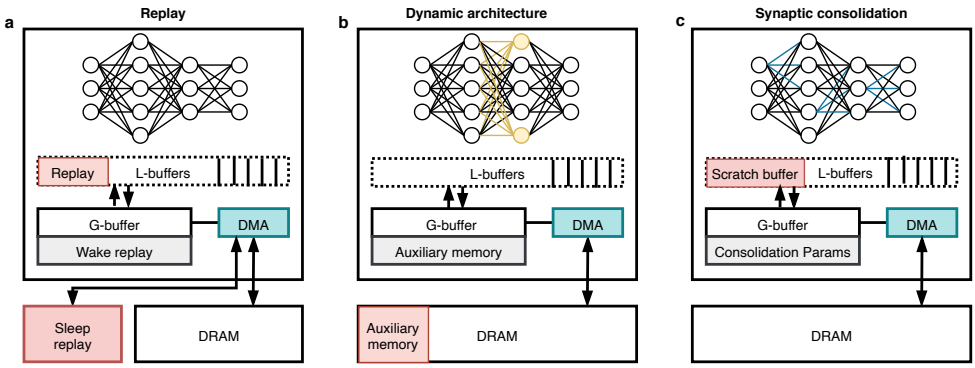

Fig. 3 Memory models. Potential memory models to enable lifelong learning features in AI accelerators, for methods such as: replay (**a**), structural plasticity (**b**), and synaptic consolidation (**c**). Heterogeneous and adaptive memory architectures that support data access with variable latency, high bandwidth, flexible data storage with minimal multi-tenancy (time lapse between two sequential tasks) will play an important role in these accelerators.

techniques to reduce the cost of individual memory access. The major component of energy consumption during memory access operation is due to data communication rather than data access; the former can incur up to $\sim 99\%$ of total energy consumption [68]. This has motivated designers to reduce the communication distance between memory and processing. A key technique in this regard is processing in memory which performs computation within the memory array. A similar technique is near-memory computation, which brings compute cores closer to memory. DLUX [43] which is an accelerator that leverages near-bank computation in 3D memory has shown on average $42\times$ improvement in *energy efficiency* on representative data centre training workloads compared to the Tesla V100 GPU. BrainChip's Akida [53] and Intel's SNN chip [57] adopt near-memory computation with distributed compute cores where each core is assigned dedicated on-chip memory.

Another way to reduce energy consumption due to memory access is to reduce the number of accesses altogether. To this end, dataflow and data layout in memory can be optimised to maximally coalesce memory access and reuse data [44]. Another technique is to use recomputation to save memory access [69]. In this technique, the intermediate states necessary for the backward pass are recomputed from the checkpoint rather than stored in memory. These are some of the prominent techniques that help tackle the *memory overhead* for on-device training.

Lifelong learning methods that mitigate catastrophic forgetting naturally exacerbate *memory overhead* during training. A conceptual memory organisation for the three lifelong learning methods is shown in Fig. 3. Structural plasticity and synaptic consolidation require additional storage on-device for the network parameters and structural pointers. In structural plasticity, the number of weights, activations, and gradient information that needs to be stored will grow over a system's lifetime, hence the *memory overhead* can linearly increase with the number of tasks encountered in the models lifetime [70]). In Fig. 3, we show this overhead with additional auxiliary

memory that can be kept in sleep mode when not in use. Synaptic consolidation often involves using additional states to represent information such as metaplasticity, consolidated weights, or probabilistic information. This can lead to up to $\sim 3.5\times$ *memory overhead* for a short sequence of tasks [13]. Moreover, consolidation approaches may require scratchpad memory to temporarily store network activity or gradient information to determine the importance of parameters. Lastly, in replay methods, in addition to storing network information, there may be a need to continuously store prior experiences in some form of memory buffer or at least some form of knowledge representing prior experiences from which synthetic data can be generated. It can lead to up to $2\times$ memory storage overhead in addition to the buffers [70], which may increase with the addition of newer tasks leading to unbounded memory requirements. The optimum size of the replay buffer is usually chosen based on the performance in the problem being solved and the resource constraints. While simpler tasks may require storing a small percentage of data (1%), more complex tasks may need more storage for acceptable performance [71]. Depending on the sample size, frequency of access and the stage of replay (sleep or wake), this data can be located in higher levels of memory hierarchy or in buffers that are close to the compute substrate, as shown in Fig. 3. In practical scenarios, a combination of methods will be chosen that can compound the *memory overhead*. It is worth emphasising that the cost of the overhead is largely determined by the location of the data in the memory hierarchy, rather than its amount. Since fetching data from an off-chip DRAM can incur $\sim 33\times$ more energy compared to on-chip global buffer [61], even a small overhead can incur substantial energy cost as it resides in higher levels of memory. Consequently, memory optimisation is critical for most existing lifelong learning algorithms on untethered devices.

Dynamic interconnection networks

The interconnect topology and the speed of the link can critically impact the flexibility, functionality, and real-time performance of an accelerator. Topologies such as mesh, tree, ring, and hybrid are commonly used to support global connectivity, especially in AI accelerators, where common patterns are observed in weight updates and underlying communication. The selection of topology and packet-based communication mode (unicast, multi-cast, or broadcast) directly depends on factors such as latency, peak and average bandwidth, and traffic patterns. Common NoC topologies are variants of the H-Tree, hierarchical tree, and toroidal mesh [47, 50, 52, 72]. The 2D torus topology offers high throughput as it enables equal bisection bandwidth between the two halves of the network [41, 64, 73] and supports parallel synchronous and asynchronous stochastic gradient descent operations. Additional schemes such as population-based hierarchical connectivity introduced in Loihi [50], can reduce the connectivity resources by an order of magnitude by utilising predefined connectivity templates mapped to specific populations.

To support features such as neuronal pruning and neurogenesis which are dynamically triggered by novel input stimuli, it is important to select a topology that is reliable, scalable, and offers fine-grained routing reconfigurability. Rapid reallocation or generation of new resources is required during high periods of activity. Full connectivity to all the processing elements is desirable, as new neurons can be generated in

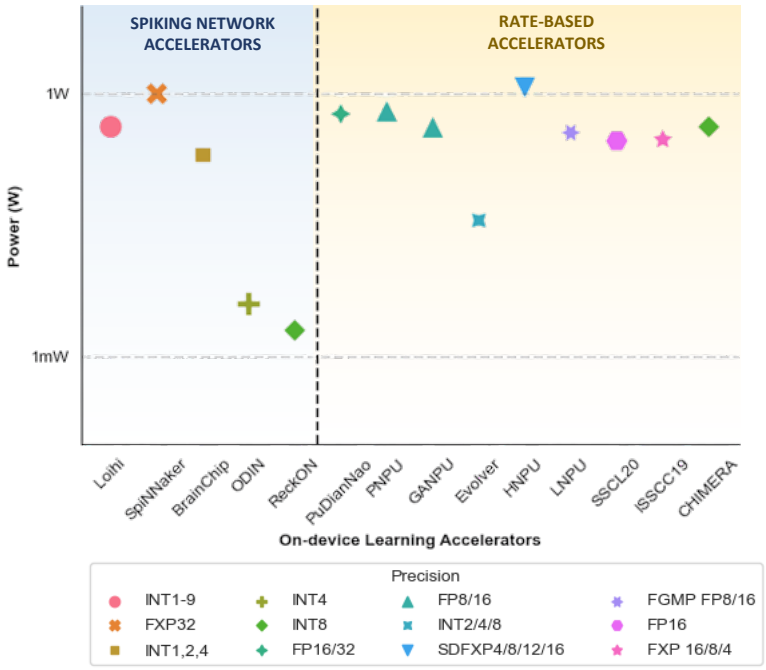

Fig. 4 Maximum accelerator power with respect to quantisation and numerical formats. In general, accelerators operating on lower bit precision (< 8-bit) demonstrate lower power consumption.

any layer. At the same time, greater routing feasibility is preferred at the local nodes as potential synapses are formed. Furthermore, fast router links for global and local connectivity will be required to facilitate real-time updates.

It is important to note that support for the above mentioned features comes at the cost of increased node count and, therefore, higher power and area - undesirable for untethered devices. For instance, full connectivity leads to large-area and poor scalability of the models. On the other hand, fast router links such as high bandwidth metallic interconnects and advanced digital routers are power hungry (for example, 2GHz on-chip network router fabricated at 16nm CMOS process consumes 2.4 to 2.8W of power [74]). Optical interconnects or hybrid optical interconnects (2D or 3D topologies) can be potential solutions in the long term [75, 76].

Quantisation

Quantisation can offer significant benefits when operating under resource constraints, enabling the reassignment of resources and increasing the feasibility of model checkpointing. For example, quantising model parameters from 32-bit to 8-bit, can reduce the model size by 50%–75% and reduce the energy consumption by 67%–75% [77]. Fig. 4 and Tables 2 and 3 show how different accelerators use quantised precision to achieve low energy and high throughput [44, 47, 50, 55, 58, 72, 78, 79]. However,

also note that few of the accelerators shown in Fig. 4 demonstrate significantly lower power due to other factors such as chip size and on-chip memory (refer to Tables 2 and 3) for details.

Existing approaches for minimisation of quantisation error fall under three categories: quantisation-aware training, post-training quantisation, and fully quantised training. Quantisation-conscious training approaches train or fine-tune models with simulated quantisation operations to learn quantised weights and activations. On the other hand, no training is involved in the post-training quantisation approach; the high-precision parameters are directly converted to low-precision parameters. While the former two approaches are used for inference applications, fully quantised training can be used for training a model that quantises gradients in addition to weights and activations [80]. Quantisation on hardware generally involves reduction of bit-width and designing custom processing elements to operate the optimised bit-precision. For example, studies have implemented a 2xfp 8/1xfp16 (fp - floating point) configurable fused-multiply add [44]; some techniques removed fp special values such as *NaN, Inf* and *subnormal* during training stage [81], and some newer works suggest near-memory-processing techniques [82, 83] wherein simple operations are performed by data with high-bit precision in external memory and the data is fetched to the accelerator after quantisation in the memory.

Lifelong learning models often require high-precision computation and are challenging to operate in a low-precision regime. For example, in regularisation techniques, previous knowledge is protected by reducing the changes to the weights that are deemed important. However, low-precision weights prohibit such precise changes in magnitude, thus requiring longer duration to learn a task. Another challenge is that these models require multiple quantisation levels for the different lifelong learning methods. When these models are deployed on accelerators incorporating energy-efficient MACs, the challenge exacerbates due to the dynamic quantisation requirements on-device. Therefore, the system should have knowledge of the statistical parameters to identify the right quantisation mechanism. However, these mechanisms have a large search space to narrow down to an optimal quantisation level. Studies suggest that quantisation can be used to improve model fit on-device and throughput, where these systems carry out continuous online training with training frequencies from *an hourly to daily* basis and training duration between *seconds to days* [84]. Other techniques, such as adaptive quantised training [85, 86], show promise in their applicability to lifelong learning accelerators. We can assess the goodness of a quantisation approach by performing ablation studies to determine how byte size affects the *arithmetic intensity* and thereby throughput for real-time learning. Additional insights can be gained by studying the *memory footprint* for a specific type of quantisation.

Sparsity

Recent studies on ML models show that sparsification can lead to a reduction in model size by 1–2 orders of magnitude [87]. In sparsification methods, we reduce representational complexity using only a subset of the features (eliminating ineffectual computations and reducing storage by encoding only nonzero values) and achieve the corresponding improvements in computational, storage and *energy efficiency* without

loss of accuracy. Unlike mainstream over-parameterized models that tend to overfit to the data and degrade generalisation to unseen examples, sparsity can help with generalisation. Sparsification can be either structural (*i.e.* weights, neurons, heads) or ephemeral (*i.e.* activations, gradients, errors). Generally, accelerators aim for either sparse matrix-vector or sparse matrix-matrix multiplications. Sparsity is inherently unstructured (*e.g.*, activations, gradients), where the *nonzero* elements are randomly scattered. Often to improve hardware execution, the structure is introduced through model operators or weight pruning in coarse-grain blocks. Most accelerators utilise co-design approaches of a sparse training algorithm and hardware. Specific approaches include structural weight sparsity by storing weights in a compressed block format, supporting the arbitrary reuse of matrices or their elements, sparse compression with event-driven activity gating, using address event representation and using blocked bitmap storage to implement sparse vector products [44, 46, 47, 50, 55, 65, 66]. A sparse matrix storage format determines ranges of sparsity where it performs most efficiently. Additionally, spiking accelerators incorporate the aforementioned optimisations since SNNs fundamentally feature a high degree of sparseness in their activity [50, 52, 56]. Tables 2 and 3 show how different accelerators handle sparse computations.

Sparsity can be quite unstructured and fine-grained in lifelong learning methods. Dynamic sparsity techniques that iteratively combine pruning and regrowth of elements during the training process, as in [16, 88], will be more suitable than static sparsity approaches that do not account for model structure updates during training. Techniques such as ephemeral sparsity during training, sparsification during training, and dully sparse training can be utilised based on the method. Structural plasticity methods introduce load imbalance depending on the distribution of zeros across different processing elements (*i.e.* inter-PE and intra-PE load imbalances). This requires a dedicated block that can share the workload at run-time and support asynchronous execution. It seems that the optimisation space for these methods is quite large. Algorithms can guide the types of sparsity that can be supported. A process like neurogenesis can be valuable for sparse high-dimensional representations that remain stable with adaptation [89]. Several of the hardware metrics such as *memory footprint, learning cost, power efficiency* are directly impacted by the level of sparsity, extraction of non-zeros, data decoding, and synchronisation. Furthermore, *communication overhead* captures the peak utilisation of accelerators' computational resources and off-chip bandwidth.

Programmability

Any physical accelerator is designed within fixed resource and energy budgets. Programmability allows a system to perform flexible operations within a fixed budget. However, an inverse relationship is observed between the programmability and *energy efficiency* of the accelerators [90]. Prior studies have focused on improving the programmability of the accelerators to support multiple learning algorithms and architectures within a power budget of $< 1W$ [50, 91, 92]. For example, researchers have explored different techniques such as using custom instruction set architectures [49, 73], multi-core architectures (large core counts) utilising smart interconnects with each

core capable of executing completely distinct programs [93] and support for dynamic hardware reconfiguration [46, 64, 72, 94].

To perform lifelong learning in the wild (in untethered scenarios) [22], systems may have changing goals and functionalities, utilise different learning rules, or even different models and architectures. Translating this to accelerator design requires flexibility in selection of programmable methods to update a model's architecture and parameters. Such programmability is crucial for reassigning resources under SWAP constraints where several features are necessary but need not be active simultaneously. For example, a system could periodically switch between employing structural plasticity or complex synapses and replay. In this manner, the advantages of different techniques can be utilised based on the needs at run-time. Additionally, within the paradigm of lifelong learning, highly programmable accelerators are expected to support algorithms that evolve over time, enabling efficient implementations of structural plasticity.

General design considerations

Following Occam's razor, designing lifelong learning accelerators can be seen as a form of multi-scale optimisation and may effectively increase the system robustness and reconfigurability. The greatest difficulty lies in activating all of the above features to their full potential in a single accelerator, so that it can be used for lifelong learning.

While lifelong learning accelerators feature the real-time learning capabilities of on-device learning accelerators, the former require quite a few additional features. In particular, lifelong learning accelerators must optimise mapping of sparse and irregular data movement within a limited power budget. Concretely, this optimisation supports structural changes during the model lifecycle. As far as memory is considered, a major difference is in the dynamic memory structures. Memory can grow or shrink at multiple abstractions (*e.g.*, temporary buffers, replay buffers, plasticity modules) based on the lifelong learning mechanism. On the other hand, on-device learning accelerators can experience model changes, but the memory access patterns might not change across inference or training. Reconfigurability is a key element in any accelerator, but for lifelong learning systems, a more fine-granular reconfigurability is necessary to direct resources when rapid plasticity occurs at run-time. In recent times, most machine learning models use compression techniques such as sparsity and quantisation. What makes it difficult in the context of lifelong learning is that the early stage structure adaptation and later stage adaptation can be varied, or there can be unstructured sparsity caused by neurogenesis. This means the system has to perform adaptive compression, whether it is quantisation or other forms of compression. New tapered precision numerical formats that are effective during training and inference might come to rescue [95, 96]. A critical point is that all of these methods should address the extreme SWaP requirements associated with untethered applications.

In general, the surge in different algorithmic mechanisms and the AI models, and evolving evaluation methods cause a widespread in the accelerator designs. This makes it hard to determine the state-of-the-art and make direct comparisons on performance and order-of-magnitude improvements. For example, if [33] serves as a reference for

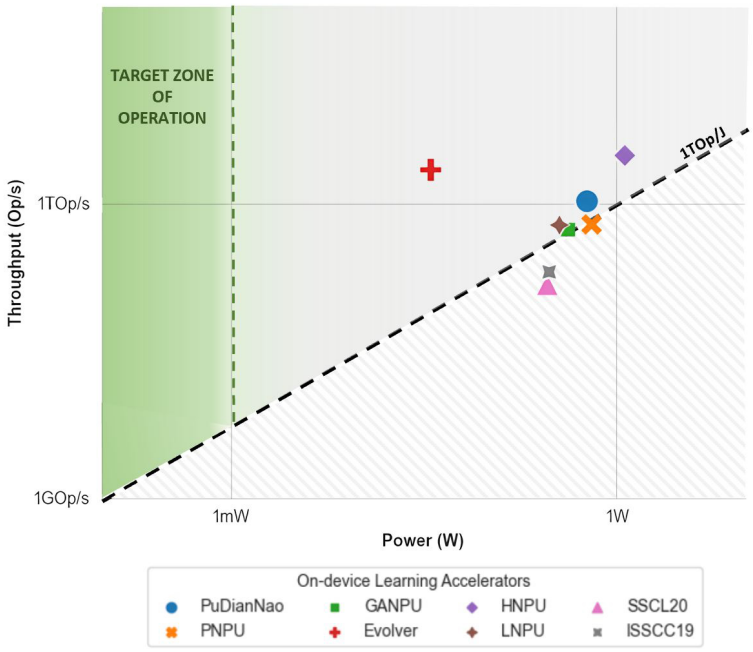

Fig. 5 Accelerator power vs. throughput. We observe the power vs. throughput for accelerators that support on-device training in untethered environments [44, 46–48, 62, 72, 78, 91]. The axes are set in log scale. The green region represents the target zone of operation, ($< 1mW$ and $> 1GOp/s$), for lifelong learning edge accelerators [100].

lifelong learning accelerators, one can state the goal for the accelerators is to operate at 500 MOP/s throughput and $\sim$ 10-15 ms of training latency per sample with 14mW power at 14nm node. However, it is important to note that [33] currently supports regularisation-based mechanisms and is also not rigorously optimised. This might not then be a fair baseline for accelerators that support all three mechanisms or a different class of mechanisms [97, 98], as the architectures can lean towards compute-bound or memory-bound based on the algorithmic mechanism.We intend to revisit the discussion of lifelong learning on spiking neural network accelerators as they mature.

Future designs for lifelong learning accelerators

The tinyML community [99] has specified a set of targets for edge AI accelerators: power consumption <1mW, numerical precision of $\leq$4-bits, expected best case latency in the range 1-10ns, SRAM capacity of 100kB-150kB, and DRAM capacity of 0.5MB-1MB. Fig. 5 illustrates the power and throughput values of a subset of current AI accelerators, with on-device learning capabilities [44, 46–48, 62, 72, 78, 91]. As can be seen, we still have a ways to go before reaching the targets set for inference accelerators. For instance, the power consumption of most of the current on-device learning accelerators ranges from 36mW to 1.16W, throughput ranges from 0.15 TOP/s to 3

TOP/s, and numerical precision is between 2 and 16-bits (integer and floating point) albeit a couple of exceptions [Tables.2 & 3].

Simultaneously satisfying all TinyML design criteria may be challenging in the near future, without architectural innovations, aggressive optimisations, and new circuit designs. This challenge is further exacerbated for lifelong learning machines. For a lifelong learning accelerator, at the edge, we propose that the target zone of operation is < 1mW and $> 1GOp/s$, which is similar to the neuromorphic system counterparts. Previous studies have identified this zone of operation for the extreme edge [100]. This zone is suitable for lifelong learning applications in untethered environments, given operational constraints. It should be noted that this zone is an initial target and can be adapted as there is more development in the models and accelerators. Although optimisations in current digital accelerators can extend their capabilities, it is important to rethink accelerator architectures for lifelong learning.

Reconfigurable architectures

High degree of reconfigurability is required primarily for structural plasticity mechanisms, as it supports the physical changes in the model primitives. To provide such fidelity, design aspects to consider include : i) a shared bus with flexible bandwidth to interact amongst all layers (*e.g.*- neurogenesis/pruning); ii) presence of dynamic precision accumulators or integrators to circumvent rapid saturation and excessive neuron firing; and iii) distributed memory (SRAMs), that can be tuned to different sleep modes when unused for timing-sensitive routines (2MB SRAM fabricated in 65nm process can result in power saving of $\sim 5000\times$ when running in deep-sleep mode as compared to active mode [101]). Communication schemes in which the synaptic connections are virtually formed or pruned, such as address-event representation (AER) [102], enhanced AER [103], or synthetic synapse representation (SSR) [104], are potential pathways to realise a shared bus with flexible bandwidth. New tapered numerical formats, such as Posit [105], can be used in the service of dynamic precision modulators. These formats have shown potential to operate with high accuracy and at >8-bit precision in both inference and training operations [96, 106].

To address reconfigurability in cases where the parameters of a cluster of neurons are regularly tuned (*e.g.*: metaplasticity), time multiplexing [107] can be considered. One approach is for the processing elements (PEs) to be time multiplexed such that each PE serves multiple neurons, namely one-serve-multiple. This can facilitate neurogenesis or synaptogenesis within preset computational resources. In this approach, additional memory is required to account for the new connections created during structural plasticity. Moreover, the memory overhead problem is aggravated due to the requirement to store neurons with multiple internal states (*e.g.*-sigmoid/ReLU as opposed to spiking neurons/LSTM cells). In other scenarios where fixed neurons are added, the memory constraint is no longer present and PEs can be equipped with linear-feedback shift registers (LFSRs) to formulate synaptic pathways (as in SSR) or generate the synaptic weights [108]. Rather than storing synaptic weights, generating them can improve *energy efficiency* and give the accelerator an additional degree of freedom to create or prune synaptic connections. When forming and pruning synapses,

the model must be aware of the importance of a synaptic connection to the previously learnt information.

Memory topologies

For memory design, there are two main design considerations : i) high data bandwidth for rapid learning and ii) large memory footprint.

Addressing the first design consideration is particularly challenging, as data transfer across the chip can incur high energy (up to 62% of total energy in a mobile device [109]), which limits the bandwidth that can be supported in energy-constrained platforms. We envision memory organisation biased towards local computation as a promising approach to address this issue. Although current processing-in-memory architectures leverage this principle, these allow minimal modification in memory arrays, so it may prove difficult to accommodate complex computations. Developments to support such computations can prove impactful to realise energy-efficient lifelong learning accelerators. In this regard, processing-near-memory architectures have more potential, since the logic layer being close to memory can offer more flexibility. In addition, advances in memory technology are also necessary to support high bandwidth data transfer in energy-constrained platforms. Newer technologies, such as optical interconnects, can be incorporated in 3D memory to improve memory bandwidth and *energy efficiency*. More fine-grained DRAM architectures are also being explored that activate a smaller portion of the array and reduce the communication distance between I/O and cell array for *energy efficiency* [110, 111]. Progress on this front can also prove beneficial in supporting high memory bandwidth in lifelong learning accelerators.

The second design consideration of large memory footprint motivates us to look toward software-controlled heterogeneous memory for lifelong learning accelerators. In traditional hardware-managed memory hierarchy, high-speed memory incurs a large chip area without contributing to memory capacity, which is not feasible in area-constrained platforms. This issue can be avoided with software-controlled memory. It can optimally use heterogeneous memory to enhance on-chip memory capacity by allocating data with different lifetimes and access frequencies to different types of memory. For example, the gradient/ network activity information required in consolidation mechanisms can be stored in scratchpad memory or any memory technology with low access time and higher endurance while network parameters that are updated on a slower timescale, for example, parameter importance metrics, can be stored in a compact slower memory. Replay methods may require storing a substantial number of samples from previous tasks, but since these samples are usually interleaved with current task samples, their access may be less latency-sensitive. A more compact memory technology with higher access latency such as HBM or HMC could be more suitable for storing them. With advances in emerging memory technologies (more on this in subsection Beyond CMOS and emerging technologies), designers can choose from an array of devices from memory to storage to address the *memory overhead* associated with lifelong learning methods. However, it remains to be understood how memory should be distributed to optimally support data access patterns and *memory overhead* of the different lifelong learning methods.

On-chip communication

In the context of lifelong learning, the Network on Chip (NoC) should be able to meet the varying data and bandwidth requirements without degradation in throughput, as the network grows in real-time and its structure is altered. On-chip communication should arguably promote reliability and availability. Time-division-multiplexed communication (supporting asynchronous and synchronous solutions) [112, 113], and dense 3D networks can offer interim solutions to enable multiple tiers of interconnections for structural plasticity mechanisms and regularisation techniques with high bandwidth communication. It is important to note that the lack of high bandwidth on-chip communication has a direct impact on the performance of lifelong learning accelerators, such as slow adaptation for neurogenesis, throughput limitations for replay methods, missing critical information during asynchronous updates, and executing time-critical updates in regularisation while learning.

Other approaches to improve on-chip communication is to use intra-layer optical interconnects, where a single transmitter delivers data to multiple arbitrarily arranged receivers to support local learning [76]. Optical wiring and modulators can enable direct photonic broadcast (many to many, without a need for time-multiplexed communication) with minimum latency and energy consumption [76], or point-to-point communication via optical superchannels (point-to-point communication via optical superchannels uses upto 1Tb/s electro-optical modulator and can offer $\sim$13$\times$ higher bandwidth than advanced digital routers) [74, 114].

Beyond CMOS and emerging technologies

When we contemplate the potential of technologies beyond CMOS and those that are emerging, for lifelong learning, we can separate them into two distinct categories.

The first category relates to how the inherent characteristics of emerging technologies that are close to maturation can address the computational challenges of lifelong learning. *Memory overhead* is one of the primary examples: lifelong learning methods incur significant *memory overhead* and should be supported within the form factor of untethered devices. These methods also require memory with a range of latency and endurance criteria to optimise performance.

Emerging non-volatile memory (NVM) such as resistive random access memory (RRAM), phase change memory (PCRAM), spin transfer torque magnetoresistive random access memory (STT-MRAM), hold promise in this regard. RRAM and PCRAM are suitable for meeting memory density requirements, as they are capable of 3D integration, which offers 3$\times$ higher density compared to contemporary DRAM [115]. STT-MRAM devices can be used to store parameters that are updated frequently, as they exhibit fast read and write time and high endurance [116]. Combined with CMOS memory technology, emerging memory devices support heterogeneous architectures with a range of latency, density, and endurance characteristics that can meet the requirements of lifelong learning methods. Mature RRAM and PCM devices can also be incorporated into processing-in-memory (PIM) architectures to support streaming input scenarios. It is projected that RRAM-based PIM architecture consumes $\sim$ 57% less energy than SRAM [117]. However, device engineering is required to make them

operate at lower technology nodes. Moreover, the devices exhibit non-ideal characteristics such as nonlinearity in tuning, low precision, and limited endurance, which can negatively impact the model performance. These challenges call for further research to improve device characteristics along with creative adaptation of device variability in algorithms and architectures that alleviate the effects of non-idealities.

The second category relates to how the needs of lifelong learning architectures open up the design space of emerging devices and materials. For example, device properties, such as the ability to modulate the change in internal state at different voltage or current levels, can be leveraged to implement metaplasticity mechanisms [118, 119]. Likewise, a performance analysis of memristive devices for online learning has shown that tolerances in the write error of ≈40% can be achieved due to the self-correcting nature of supervised learning algorithms and synaptic plasticity models [120]. This potentially opens up the design space to devices that may exhibit a lower degree of precision or control of resistive states. Devices with different degrees of retention can contribute to lifelong learning systems: high-retention devices can be used to emulate long-term network parameters, whereas low-retention devices can be leveraged for short-term metaplasticity. Frequent updates may require devices with higher endurance in both scenarios.

It is important to highlight the scarcity of studies that evaluate the potential of emerging technologies in the context of lifelong learning [119]. A fundamental question to be explored is what requirements these devices should have to maximise their potential for lifelong learning. In fact, it is plausible that different parts of an architecture may require different device characteristics. All these factors open opportunities for research on device, architecture, and algorithm co-design, to leverage beyond CMOS technologies effectively for lifelong learning on untethered platforms.

Outlook

For AI models to learn efficiently and continually in practical settings, careful consideration of the underlying AI hardware accelerators is required. Rapid innovations are occurring in both lifelong learning models and the underlying hardware. Therefore, model and hardware efficiencies are intertwined, and co-design approaches are essential in order to design good machines. To facilitate co-design between model and hardware, we have outlined features for lifelong learning accelerators that are agnostic or tied to specific methods. These methods serve as the backbone for most existing lifelong learning models. However, as model complexity increases, transitioning from current AI accelerators to the next generation of accelerators will be necessary for both advances in research and practical applications.

We also explored existing accelerators for learning at the edge, and highlighted how lifelong learning can exacerbate the gap between an ideal accelerator and existing AI accelerators. We examined how common optimisation approaches can be adapted to focus on features necessary for lifelong learning, providing a route for existing AI accelerators to become applicable to lifelong learning. Often these optimisations will be application specific depending on the models tried and tested in their corresponding domains prior to deployment. To design a more general-purpose lifelong learning

accelerator, the system should have the ability to autonomously self-tune at a finer granularity based on the application, and have the capability to evaluate the trade-offs across the learning features. However, these insights are limited to the scope of existing accelerators, and we also discussed areas that have room for further exploration, particularly beyond CMOS and emerging technologies.

What then is the optimal solution for lifelong learning accelerators? As a starting point, this can include achieving on-device training for high-dimensional and large-scale problems on the cheap, fine granular run-time reprogrammability of compute/memory resources and novel heterogeneous memory architectures that support high-density storage and fast access with increased bandwidth. Moreover, all of these features should be supported in sub-milliWatt power budget so that they can operate seamlessly in untethered environments. We also provided an initial set of metrics that can help assess the different solutions in terms of performance and cost. Although there is unlikely to be one unified metric to clearly identify if one solution is better than another, the metrics discussed should offer the granularity to compare accelerators based on applications.

The brain's ability to adapt throughout its lifetime makes it an attractive source of inspiration for lifelong learning. However, the complexity of biological learning can be overwhelming. Therefore, it can be helpful to identify key high-level neuroscience insights when considering future directions for the development of lifelong learning accelerators.

First, the brain has a considerable diversity of learning mechanisms spanning a range of spatial and temporal scales and the specific types of plasticity can vary considerably between brain regions and between different types of computational tasks. For example, while a structural plasticity mechanism such as neurogenesis may be useful for domains with continuously evolving input and output dimensionalities, such as episodic memory, it may not be necessary for systems where the input and output dimensionalities are well defined, such as motor control or vision processing [121].

Second, related to the task specificity of the learning mechanisms, the brain often dynamically modifies the plasticity itself through neuromodulation or feedback circuits. The use of such top-down mechanisms may control computational processing according to different behavioural contexts, but it can also select which plasticity mechanisms are active in the system.

Third, sleep/wake cycles enable the system to switch between encoding and consolidation modes, and permit the transfer of information between brain regions. Such modulatory control allows a system to differentially regulate what information is learnt.

Furthermore, when considering the future development of lifelong learning accelerators, a number of key challenges and open questions remain. To start, how do we co-design hardware architectures to achieve high degrees of run-time reconfigurability? And what are the associated trade-offs between recovery time, cost, energy, and mean accuracy? Can new architectures support structural changes at fine granularity in order to realise potential synapses? What is the relationship between synaptic plasticity, structural plasticity, learning dynamics, and generalisation? How should memory

be distributed among different technologies or topologies to reduce the memory overhead of lifelong learning? How best to use emerging devices with various ranges of read-write latency/energy and endurance to optimally support the memory requirements of lifelong learning? And what should be the ideal distribution between existing and emerging memory technologies?

Acknowledgments. We thank members of the Neuromorphic AI lab, Peter Helfer, Tej Pandit, Vedant Karia, and S. Hamed Fatemi Langroudi, for discussions on this topic. We also thank the reviewers for their valuable feedback. Part of this material is based on research sponsored by Air Force Research Laboratory under agreement number FA8750-20-2-1003 through BAA FA8750-19-S-7010. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government. This article has been approved for public release; distribution unlimited (Case No. AFRL-2023-3120, 28 Jun 2023). AY acknowledges Laboratory Directed Research and Development (LDRD) funding from Argonne National Laboratory, provided by the Director, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-06CH11357.

Declarations

Competing interests. The authors declare the following competing interests: The authors declare no competing interests.

Author Contributions. DK led the team on the design and concept for the manuscript. DK, AD, AZ, FZ, JA, AYG, NS, EN, MM, VL, CT and BE had multiple rounds of discussions in conceptualization of the manuscript and have contributed to the iterative draft manuscripts and the main manuscript text. AD and AZ prepared the figures, with input from DK, FZ, and, NS. AD, FZ and AZ collected data and prepared the tables in the manuscript. DK revised it critically for important intellectual content. All authors commented on the manuscript and reviewed the final-version of the manuscript.

References

[1] Kudithipudi, D., Aguilar-Simon, M., Babb, J., Bazhenov, M., Blackiston, D., Bongard, J., Brna, A.P., Chakravarthi Raja, S., Cheney, N., Clune, J., *et al.*: Biological underpinnings for lifelong learning machines. Nature Machine Intelligence **4**(3), 196–210 (2022)

[2] Delange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A Continual Learning Survey: Defying Forgetting in Classification Tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)

[3] Thrun, S., Mitchell, T.M.: Lifelong robot learning. Robotics and autonomous systems **15**(1-2), 25–46 (1995)

[4] McCloskey, M., Cohen, N.J.: Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In: Psychology of Learning and Motivation vol. 24, pp. 109–165. Elsevier, ??? (1989)

[5] McClelland, J.L., McNaughton, B.L., O'Reilly, R.C.: Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. Psychological review **102**(3), 419 (1995)

[6] Pratt, L.Y., Mostow, J., Kamm, C.A., Kamm, A.A., *et al.*: Direct transfer of learned information among neural networks. In: Aaai, vol. 91, pp. 584–589 (1991)

[7] Caruana, R.: Multitask Learning. Machine learning **28**(1), 41–75 (1997)

[8] Fei-Fei, L., Fergus, R., Perona, P.: One-Shot Learning of Object Categories. IEEE transactions on pattern analysis and machine intelligence **28**(4), 594–611 (2006)

[9] Thrun, S., Pratt, L.: Learning to Learn: Introduction and Overview. In: Learning to Learn, pp. 3–17. Springer, ??? (1998)

[10] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 201611835 (2017)

[11] Zenke, F., Poole, B., Ganguli, S.: Continual Learning through Synaptic Intelligence. In: International Conference on Machine Learning, pp. 3987–3995 (2017). PMLR

[12] Laborieux, A., Ernoult, M., Hirtzlin, T., Querlioz, D.: Synaptic metaplasticity in binarized neural networks. Nature communications **12**(1), 1–12 (2021)

[13] Soures, N., Helfer, P., Daram, A., Pandit, T., Kudithipudi, D.: TACOS: Task Agnostic Continual Learning in Spiking Neural Networks. In: Theory and Foundation of Continual Learning Workshop at ICML'2021 (July 2021)

[14] Schug. Simon, S.A. Benzing. Frederik: Task-Agnostic Continual Learning via Stochastic Synapses. https://drive.google.com/file/d/1ZQg7Lb8IoVdHet3AwQFWTvg5QBUsbZJX/view. (Accessed on 04/15/2022) (2020)

[15] Ebrahimi, S., Meier, F., Calandra, R., Darrell, T., Rohrbach, M.: Adversarial

Continual Learning. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16, pp. 386–402 (2020). Springer

[16] Pandit, T., Kudithipudi, D.: Relational Neurogenesis for Lifelong Learning Agents. In: Proceedings of the Neuro-inspired Computational Elements Workshop, pp. 1–9 (2020)

[17] Masse, N.Y., Grant, G.D., Freedman, D.J.: Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. Proceedings of the National Academy of Sciences **115**(44), 10467–10475 (2018)

[18] Rebuffi, S., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCaRL: Incremental Classifier and Representation Learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition. CVPR'17, pp. 5533–5542 (2017)

[19] Lopez-Paz, D., Ranzato, M.: Gradient Episodic Memory for Continual Learning. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS'17, pp. 6470–6479. Curran Associates Inc., USA (2017)

[20] Ven, G.M., Siegelmann, H.T., Tolias, A.S.: Brain-inspired replay for continual learning with artificial neural networks. Nature communications **11**(1), 1–14 (2020)

[21] Hayes, T.L., Krishnan, G.P., Bazhenov, M., Siegelmann, H.T., Sejnowski, T.J., Kanan, C.: Replay in deep learning: Current approaches and missing biological elements. Neural Computation **33**(11), 2908–2950 (2021)

[22] Mundt, M., Hong, Y., Pliushch, I., Ramesh, V.: A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. Neural Networks **160**, 306–336 (2023)

[23] Kwon, Y.D., Chauhan, J., Kumar, A., HKUST, P.H., Mascolo, C.: Exploring System Performance of Continual Learning for Mobile and Embedded Sensing Applications. In: 2021 IEEE/ACM Symposium on Edge Computing (SEC), pp. 319–332 (2021). IEEE

[24] Ven, G.M., Tolias, A.S.: Three scenarios for continual learning. arXiv preprint arXiv:1904.07734 (2019)

[25] Gupta, V., Narwariya, J., Malhotra, P., Vig, L., Shroff, G.: Continual Learning for Multivariate Time Series Tasks with Variable Input Dimensions. In: 2021 IEEE International Conference on Data Mining (ICDM), pp. 161–170 (2021). IEEE

[26] Seshia, S.A., Sadigh, D., Sastry, S.S.: Toward verified artificial intelligence.

Communications of the ACM **65**(7), 46–55 (2022)

[27] Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A.A., Pritzel, A., Wierstra, D.: Pathnet: Evolution channels gradient descent in super neural networks. arXiv preprint arXiv:1701.08734 (2017)

[28] Lee, S., Ha, J., Zhang, D., Kim, G.: A Neural Dirichlet Process Mixture Model for Task-Free Continual Learning. arXiv preprint arXiv:2001.00689 (2020)

[29] Harris, M.: Inside Pascal: NVIDIA's Newest Computing Platform. https://developer.nvidia.com/blog/inside-pascal/ (2016)

[30] Norrie, T., Patil, N., Yoon, D.H., Kurian, G., Li, S., Laudon, J., Young, C., Jouppi, N., Patterson, D.: The Design Process for Google's Training Chips: TPUv2 and TPUv3. IEEE Micro **41**(2), 56–63 (2021)

[31] New, A., Baker, M., Nguyen, E., Vallabha, G.: Lifelong Learning Metrics. arXiv preprint arXiv:2201.08278 (2022)

[32] Zohora, F.T., Karia, V., Daram, A.R., Zyarah, A.M., Kudithipudi, D.: Metaplasticnet: Architecture with Probabilistic Metaplastic Synapses for Continual Learning. In: 2021 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5 (2021). IEEE

[33] Karia, V., Zohora, F., Soures, N., Kudithipudi, D.: SCOLAR: A Spiking Digital Accelerator with Dual Fixed Point for Continual Learning. In: 2022 IEEE International Symposium on Circuits and Systems (ISCAS) (2022). IEEE

[34] Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., Maltoni, D.: Don't forget, there is more than forgetting: new metrics for Continual Learning. arXiv preprint arXiv:1810.13166 (2018)

[35] Lesort, T., Lomonaco, V., Stoian, A., Maltoni, D., Filliat, D., Díaz-Rodríguez, N.: Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. Information fusion **58**, 52–68 (2020)

[36] Ravaglia, L., Rusci, M., Capotondi, A., Conti, F., Pellegrini, L., Lomonaco, V., Maltoni, D., Benini, L.: Memory-Latency-Accuracy Trade-Offs for Continual Learning on a RISC-v Extreme-Edge Node. In: 2020 IEEE Workshop on Signal Processing Systems (SiPS), pp. 1–6 (2020). IEEE

[37] De Lange, M., Ven, G., Tuytelaars, T.: Continual evaluation for lifelong learning: Identifying the stability gap. Proceedings ICLR 2023 (2023)

[38] Reddi, V.J., Cheng, C., Kanter, D., Mattson, P., Schmuelling, G., Wu, C.-J., Anderson, B., Breughe, M., Charlebois, M., Chou, W., *et al.*: MLPerf Inference Benchmark. In: 2020 ACM/IEEE 47th Annual International Symposium

on Computer Architecture (ISCA), pp. 446–459 (2020). IEEE

[39] Vanschoren, J., Van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: networked science in machine learning. ACM SIGKDD Explorations Newsletter **15**(2), 49–60 (2014)

[40] Davies, M.: Benchmarks for progress in neuromorphic computing. Nature Machine Intelligence **1**(9), 386–388 (2019)

[41] Jouppi, N.P., Yoon, D.H., Kurian, G., Li, S., Patil, N., Laudon, J., Young, C., Patterson, D.: A domain-specific supercomputer for training deep neural networks. Communications of the ACM **63**(7), 67–78 (2020)

[42] Chen, Y.-H., Yang, T.-J., Emer, J., Sze, V.: Eyeriss v2: A Flexible Accelerator for Emerging Deep Neural Networks on Mobile Devices. IEEE Journal on Emerging and Selected Topics in Circuits and Systems **9**(2), 292–308 (2019)

[43] Gu, P., Xie, X., Li, S., Niu, D., Zheng, H., Malladi, K.T., Xie, Y.: DLUX: A LUT-based Near-Bank Accelerator for Data Center Deep Learning Training Workloads. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **40**(8), 1586–1599 (2020)

[44] Lee, J., Lee, J., Han, D., Lee, J., Park, G., Yoo, H.-J.: 7.7 LNPU: A 25.3 TFLOPS/W Sparse Deep-Neural-Network Learning Processor with Fine-Grained Mixed Precision of FP8-FP16. In: 2019 IEEE International Solid-State Circuits Conference-(ISSCC), pp. 142–144 (2019). IEEE

[45] Han, D., Lee, J., Yoo, H.-J.: DF-LNPU: A Pipelined Direct Feedback Alignment-Based Deep Neural Network Learning Processor for Fast Online Learning. IEEE Journal of Solid-State Circuits **56**(5), 1630–1640 (2020)

[46] Tu, F., Wu, W., Wang, Y., Chen, H., Xiong, F., Shi, M., Li, N., Deng, J., Chen, T., Liu, L., Wei, S., Xie, Y., Yin, S.: Evolver: A Deep Learning Processor With On-Device Quantization–Voltage–Frequency Tuning. IEEE Journal of Solid-State Circuits **56**(2), 658–673 (2021) https://doi.org/10.1109/JSSC.2020.3021661

[47] Han, D., Im, D., Park, G., Kim, Y., Song, S., Lee, J., Yoo, H.-J.: HNPU: An Adaptive DNN Training Processor Utilizing Stochastic Dynamic Fixed-Point and Active Bit-Precision Searching. IEEE Journal of Solid-State Circuits **56**(9), 2858–2869 (2021) https://doi.org/10.1109/JSSC.2021.3066400

[48] Kim, C., Kang, S., Shin, D., Choi, S., Kim, Y., Yoo, H.-J.: A 2.1TFLOPS/W Mobile Deep RL Accelerator with Transposable PE Array and Experience Compression. In: 2019 IEEE International Solid- State Circuits Conference - (ISSCC), pp. 136–138 (2019). https://doi.org/10.1109/ISSCC.2019.8662447

[49] Chung, E., Fowers, J., Ovtcharov, K., Papamichael, M., Caulfield, A., Massengill, T., Liu, M., Lo, D., Alkalay, S., Haselman, M., Abeydeera, M., Adams, L., Angepat, H., Boehn, C., Chiou, D., Firestein, O., Forin, A., Gatlin, K.S., Ghandi, M., Heil, S., Holohan, K., El Husseini, A., Juhasz, T., Kagi, K., Kovvuri, R.K., Lanka, S., Megen, F., Mukhortov, D., Patel, P., Perez, B., Rapsang, A., Reinhardt, S., Rouhani, B., Sapek, A., Seera, R., Shekar, S., Sridharan, B., Weisz, G., Woods, L., Yi Xiao, P., Zhang, D., Zhao, R., Burger, D.: Serving DNNs in Real Time at Datacenter Scale with Project Brainwave. IEEE Micro **38**(2), 8–20 (2018) https://doi.org/10.1109/MM.2018.022071131

[50] Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S.H., Dimou, G., Joshi, P., Imam, N., Jain, S., *et al.*: Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. Ieee Micro **38**(1), 82–99 (2018)

[51] Pfister, J.-P., Gerstner, W.: Triplets of Spikes in a Model of Spike Timing-Dependent Plasticity. Journal of Neuroscience **26**(38), 9673–9682 (2006)

[52] Furber, S.B., Galluppi, F., Temple, S., Plana, L.A.: The SpiNNaker Project. Proceedings of the IEEE **102**(5), 652–665 (2014)

[53] Demler, M.: Brainchip Akida Is A Fast Learner, Spiking-Neural-Network Processor Identifies Patterns in Unlabeled Data. Microprocessor Report (2019)

[54] Nguyen, D.-A., Tran, X.-T., Iacopi, F.: A Review of Algorithms and Hardware Implementations for Spiking Neural Networks. Journal of Low Power Electronics and Applications **11**(2), 23 (2021)

[55] Frenkel, C., Indiveri, G.: ReckOn: A 28nm Sub-mm^2 Task-Agnostic Spiking Recurrent Neural Network Processor Enabling On-Chip Learning over Second-Long Timescales. In: 2022 IEEE International Solid-State Circuits Conference (ISSCC), vol. 65, pp. 1–3 (2022). IEEE

[56] Frenkel, C., Lefebvre, M., Legat, J.-D., Bol, D.: A 0.086-mm^2 12.7-pJ/SOP 64k-Synapse 256-Neuron Online-Learning Digital Spiking Neuromorphic Processor in 28-nm CMOS. IEEE transactions on biomedical circuits and systems **13**(1), 145–158 (2018)

[57] Chen, G.K., Kumar, R., Sumbul, H.E., Knag, P.C., Krishnamurthy, R.K.: A 4096-Neuron 1M-Synapse 3.8-pJ/SOP spiking neural network with on-chip STDP learning and sparse weights in 10-nm FinFET CMOS. IEEE Journal of Solid-State Circuits **54**(4), 992–1002 (2018)

[58] Dean, M.E., Daffron, C.: A VLSI Design for Neuromorphic Computing. In: 2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pp. 87–92 (2016). IEEE

[59] Chicca, E., Stefanini, F., Indiveri, G.: Neuromorphic Electronic Circuits for

Building Autonomous Cognitive Systems. Proceedings of IEEE (2013)

[60] Basu, A., Deng, L., Frenkel, C., Zhang, X.: Spiking Neural Network Integrated Circuits: A Review of Trends and Future Directions. In: 2022 IEEE Custom Integrated Circuits Conference (CICC), pp. 1–8 (2022). IEEE

[61] Chen, Y.-H., Emer, J., Sze, V.: Using Dataflow to Optimize Energy Efficiency of Deep Neural Network Accelerators. IEEE Micro **37**(3), 12–21 (2017)

[62] Yin, S., Seo, J.-S.: A 2.6 TOPS/W 16-Bit Fixed-Point Convolutional Neural Network Learning Processor in 65-nm CMOS. IEEE Solid-State Circuits Letters **3**, 13–16 (2020) https://doi.org/10.1109/LSSC.2019.2954780

[63] Lu, C.-H., Wu, Y.-C., Yang, C.-H.: A 2.25 TOPS/W Fully-Integrated Deep CNN Learning Processor with On-Chip Training. In: 2019 IEEE Asian Solid-State Circuits Conference (A-SSCC), pp. 65–68 (2019). IEEE

[64] Fleischer, B., Shukla, S., Ziegler, M., Silberman, J., Oh, J., Srinivasan, V., Choi, J., Mueller, S., Agrawal, A., Babinsky, T., *et al.*: A Scalable Multi-TeraOPS Deep Learning Processor Core for AI Training and Inference. In: 2018 IEEE Symposium on VLSI Circuits, pp. 35–36 (2018). IEEE

[65] Qin, E., Samajdar, A., Kwon, H., Nadella, V., Srinivasan, S., Das, D., Kaul, B., Krishna, T.: SIGMA: A Sparse and Irregular GEMM Accelerator with Flexible Interconnects for DNN Training. In: 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp. 58–70 (2020). IEEE

[66] Giannoula, C., Fernandez, I., Luna, J.G., Koziris, N., Goumas, G., Mutlu, O.: SparseP: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures. Proceedings of the ACM on Measurement and Analysis of Computing Systems **6**(1), 1–49 (2022)

[67] Li, J., Yan, G., Lu, W., Jiang, S., Gong, S., Wu, J., Li, X.: SmartShuttle: Optimizing Off-Chip Memory Accesses for Deep Learning Accelerators. In: 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 343–348 (2018). IEEE

[68] Dally, W.: On the model of computation: point. Communications of the ACM **65**(9), 30–32 (2022)

[69] Chen, T., Xu, B., Zhang, C., Guestrin, C.: Training Deep Nets with Sublinear Memory Cost. arXiv preprint arXiv:1604.06174 (2016)

[70] De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., Tuytelaars, T.: A Continual Learning Survey: Defying Forgetting in Classification Tasks. IEEE transactions on pattern analysis and machine intelligence **44**(7), 3366–3385 (2021)

[71] Merlin, G., Lomonaco, V., Cossu, A., Carta, A., Bacciu, D.: Practical Recommendations for Replay-Based Continual Learning Methods. In: International Conference on Image Analysis and Processing, pp. 548–559 (2022). Springer

[72] Kang, S., Han, D., Lee, J., Im, D., Kim, S., Kim, S., Yoo, H.-J.: 7.4 GANPU: A 135TFLOPS/W Multi-DNN Training Processor for GANs with Speculative Dual-Sparsity Exploitation. In: 2020 IEEE International Solid-State Circuits Conference-(ISSCC), pp. 140–142 (2020). IEEE

[73] Mayr, C., Hoeppner, S., Furber, S.: SpiNNaker 2: A 10 Million Core Processor System for Brain Simulation and Machine Learning. arXiv preprint arXiv:1911.02385 (2019)

[74] Nedbailo, Y.A., Tokarev, D.S., Shpagilev, D.I.: Designing a QoS-enabled 2 GHz On-Chip Network Router in 16nm CMOS. In: 2022 Moscow Workshop on Electronic and Networking Technologies (MWENT), pp. 1–5 (2022). IEEE

[75] Bashir, J., Peter, E., Sarangi, S.R.: A Survey of On-Chip Optical Interconnects. ACM Computing Surveys (CSUR) **51**(6), 1–34 (2019)

[76] Shastri, B.J., Tait, A.N., Lima, T., Pernice, W.H., Bhaskaran, H., Wright, C.D., Prucnal, P.R.: Photonics for artificial intelligence and neuromorphic computing. Nature Photonics **15**(2), 102–114 (2021)

[77] Krishnamoorthi, R.: Techniques for Efficient Inference with Deep Networks. Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2) (2020)

[78] Kim, S., Lee, J., Kang, S., Lee, J., Yoo, H.-J.: A 146.52 TOPS/W Deep-Neural-Network Learning Processor with Stochastic Coarse-Fine Pruning and Adaptive Input/Output/Weight Skipping. In: 2020 IEEE Symposium on VLSI Circuits, pp. 1–2 (2020). IEEE

[79] Agrawal, A., Lee, S.K., Silberman, J., Ziegler, M., Kang, M., Venkataramani, S., Cao, N., Fleischer, B., Guillorn, M., Cohen, M., *et al.*: A 7nm 4-Core AI Chip with 25.6TFLOPS Hybrid FP8 Training, 102.4 TOPS INT4 Inference and Workload-Aware Throttling. In: 2021 IEEE International Solid-State Circuits Conference (ISSCC), vol. 64, pp. 144–146 (2021). IEEE

[80] Chen, J., Gai, Y., Yao, Z., Mahoney, M.W., Gonzalez, J.E.: A Statistical Framework for Low-bitwidth Training of Deep Neural Networks. Advances in Neural Information Processing Systems **33**, 883–894 (2020)

[81] Oh, J., Lee, S.K., Kang, M., Ziegler, M., Silberman, J., Agrawal, A., Venkataramani, S., Fleischer, B., Guillorn, M., Choi, J., *et al.*: A 3.0 TFLOPS 0.62V Scalable Processor Core for High Compute Utilization AI Training and Inference. In: 2020 IEEE Symposium on VLSI Circuits, pp. 1–2 (2020). IEEE

[82] Kim, H., Park, H., Kim, T., Cho, K., Lee, E., Ryu, S., Lee, H.-J., Choi, K., Lee, J.: GradPIM: A Practical Processing-in-DRAM Architecture for Gradient Descent. In: 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA), pp. 249–262 (2021). IEEE

[83] Zhao, Y., Liu, C., Du, Z., Guo, Q., Hu, X., Zhuang, Y., Zhang, Z., Song, X., Li, W., Zhang, X., *et al.*: Cambricon-Q: A Hybrid Architecture for Efficient Training. In: 2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA), pp. 706–719 (2021). IEEE

[84] Hazelwood, K., Bird, S., Brooks, D., Chintala, S., Diril, U., Dzhulgakov, D., Fawzy, M., Jia, B., Jia, Y., Kalro, A., *et al.*: Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective. In: 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp. 620–629 (2018). IEEE

[85] Yao, Z., Dong, Z., Zheng, Z., Gholami, A., Yu, J., Tan, E., Wang, L., Huang, Q., Wang, Y., Mahoney, M., *et al.*: HAWQ-V3: Dyadic Neural Network Quantization. In: International Conference on Machine Learning, pp. 11875–11886 (2021). PMLR

[86] Zhao, S., Yue, T., Hu, X.: Distribution-Aware Adaptive Multi-Bit Quantization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9281–9290 (2021)

[87] Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., Peste, A.: Sparsity in Deep Learning: Pruning and growth for efficient inference and training in neural networks. J. Mach. Learn. Res. **22**(241), 1–124 (2021)

[88] Zyarah, A.M., Kudithipudi, D.: Neuromorphic Architecture for the Hierarchical Temporal Memory. IEEE Transactions on Emerging Topics in Computational Intelligence **3**(1), 4–14 (2019)

[89] Davies, M., Wild, A., Orchard, G., Sandamirskaya, Y., Guerra, G.A.F., Joshi, P., Plank, P., Risbud, S.R.: Advancing Neuromorphic Computing With Loihi: A Survey of Results and Outlook. Proceedings of the IEEE **109**(5), 911–934 (2021) https://doi.org/10.1109/JPROC.2021.3067593

[90] Nowatzki, T., Gangadhan, V., Sankaralingam, K., Wright, G.: Pushing the Limits of Accelerator Efficiency While Retaining Programmability. In: 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp. 27–39 (2016). IEEE

[91] Liu, D., Chen, T., Liu, S., Zhou, J., Zhou, S., Teman, O., Feng, X., Zhou, X., Chen, Y.: PuDianNao: A Polyvalent Machine Learning Accelerator. ACM SIGARCH Computer Architecture News **43**(1), 369–381 (2015)

[92] Chen, Y., Xie, Y., Song, L., Chen, F., Tang, T.: A Survey of Accelerator Architectures for Deep Neural Networks. Engineering **6**(3), 264–274 (2020)

[93] Jia, Z., Tillman, B., Maggioni, M., Scarpazza, D.P.: Dissecting the Graphcore IPU Architecture via Microbenchmarking. arXiv preprint arXiv:1912.03413 (2019)

[94] Putic, M., Venkataramani, S., Eldridge, S., Buyuktosunoglu, A., Bose, P., Stan, M.: Dyhard-DNN: even more DNN acceleration with dynamic hardware reconfiguration. In: Proceedings of the 55th Annual Design Automation Conference. DAC '18. Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3195970.3196033 . https://doi.org/10.1145/3195970.3196033

[95] Gustafson, J.: Posit Arithmetic. Mathematica Notebook describing the posit number system **30** (2017)

[96] Langroudi, H.F., Karia, V., Carmichael, Z., Zyarah, A., Pandit, T., Gustafson, J.L., Kudithipudi, D.: ALPS: Adaptive Quantization of Deep Neural Networks with GeneraLized PositS. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3100–3109 (2021)

[97] Piyasena, D., Lam, S.-K., Wu, M.: Accelerating Continual Learning on Edge FPGA. In: 2021 31st International Conference on Field-Programmable Logic and Applications (FPL), pp. 294–300 (2021). https://doi.org/10.1109/FPL53798.2021.00059

[98] Zhang, F., Yang, L., Meng, J., Seo, J.-S., Cao, Y., Fan, D.: XST: A Crossbar Column-wise Sparse Training for Efficient Continual Learning. In: 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 48–51 (2022). IEEE

[99] Warden, P., Situnayake, D.: TinyML. O'Reilly Media, Incorporated, ??? (2019)

[100] Gao, C.: Energy-efficient recurrent neural network accelerators for real-time inference. PhD thesis, University of Zurich (2022)

[101] Badodekar, N.: Power Saving with Cypress's 65-nm Asynchronous Power-Snooze™ SRAM. Cypress Semiconductor Corporation, 001–89371 (2014-2015)

[102] Mahowald, M.: VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function. PhD thesis, California Institute of Technology (1992)

[103] Goldberg, D.H., Cauwenberghs, G., Andreou, A.G.: Probabilistic synaptic weighting in a reconfigurable network of VLSI integrate-and-fire neurons. Neural Networks **14**(6-7), 781–793 (2001)

[104] Zyarah, A.M., Gomez, K., Kudithipudi, D.: Neuromorphic System for Spatial and Temporal Information Processing. IEEE Transactions on Computers **69**(8), 1099–1112 (2020)

[105] Carmichael, Z., Langroudi, H.F., Khazanov, C., Lillie, J., Gustafson, J.L., Kudithipudi, D.: Deep Positron: A Deep Neural Network Using the Posit Number System. In: 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1421–1426 (2019). IEEE

[106] Murillo, R., Del Barrio, A.A., Botella, G., Kim, M.S., Kim, H., Bagherzadeh, N.: PLAM: A Posit Logarithm-Approximate Multiplier. IEEE Transactions on Emerging Topics in Computing **10**(4), 2079–2085 (2021)

[107] Zyarah, A.M., Kudithipudi, D.: Invited Paper: Resource Sharing in Feed Forward Neural Networks for Energy Efficiency. In: 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 543–546 (2017). IEEE

[108] Zyarah, A.M., Ramesh, A., Merkel, C., Kudithipudi, D.: Optimized hardware framework of MLP with random hidden layers for classification applications. In: Machine Intelligence and Bio-inspired Computation: Theory and Applications X, vol. 9850, p. 985007 (2016). International Society for Optics and Photonics

[109] Mutlu, O., Ghose, S., Gómez-Luna, J., Ausavarungnirun, R.: A Modern Primer on Processing in Memory. In: Emerging Computing: From Devices to Systems, pp. 171–243. Springer, ??? (2023)

[110] O'Connor, M., Chatterjee, N., Lee, D., Wilson, J., Agrawal, A., Keckler, S.W., Dally, W.J.: Fine-Grained DRAM: Energy-Efficient DRAM for Extreme Bandwidth Systems. In: Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 41–54 (2017)

[111] Olgun, A., Bostanci, F., Oliveira, G.F., Tugrul, Y.C., Bera, R., Yaglikci, A.G., Hassan, H., Ergin, O., Mutlu, O.: Sectored DRAM: An Energy-Efficient High-Throughput and Practical Fine-Grained DRAM Architecture. arXiv preprint arXiv:2207.13795 (2022)

[112] Indiveri, G., Linares-Barranco, B., Legenstein, R., Deligeorgis, G., Prodromakis, T.: Integration of nanoscale memristor synapses in neuromorphic computing architectures. Nanotechnology **24**(38), 384010 (2013)

[113] Manohar, R.: Hardware/software Co-design for Neuromorphic Systems. In: 2022 IEEE Custom Integrated Circuits Conference (CICC), pp. 01–05 (2022). IEEE

[114] Rossi, S.M., Sutili, T., Souza, A.L.N.d., Figueiredo, R.C.: Electro-Optical Modulator Requirements for 1 Tb/s per Channel Coherent Systems. Journal of Microwaves, Optoelectronics and Electromagnetic Applications **20**, 823–833

(2021)

[115] Yu, S.: Semiconductor Memory Devices and Circuits. CRC Press, ??? (2022)

[116] Park, S.P., Gupta, S., Mojumder, N., Raghunathan, A., Roy, K.: Future Cache Design using STT MRAMs for Improved Energy Efficiency: Devices, Circuits and Architecture. In: Proceedings of the 49th Annual Design Automation Conference, pp. 492–497 (2012)

[117] Yu, S., Shim, W., Peng, X., Luo, Y.: RRAM for Compute-in-Memory: From Inference to Training. IEEE Transactions on Circuits and Systems I: Regular Papers **68**(7), 2753–2765 (2021)

[118] Zhu, X., Du, C., Jeong, Y., Lu, W.D.: Emulation of synaptic metaplasticity in memristors. Nanoscale **9**(1), 45–51 (2017)

[119] Zohora, F.T., Zyarah, A.M., Soures, N., Kudithipudi, D.: Metaplasticity in Multistate Memristor Synaptic Networks. In: 2020 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5 (2020). IEEE

[120] Yanguas-Gil, A.: Memristor design rules for dynamic learning and edge processing applications. APL Materials **7**(9), 091102 (2019) https://doi.org/10.1063/1.5109910 https://doi.org/10.1063/1.5109910

[121] Aimone, J.B., Deng, W., Gage, F.H.: Resolving New Memories: A Critical Look at the Dentate Gyrus, Adult Neurogenesis, and Pattern Separation. Neuron **70**(4), 589–596 (2011)

[122] Prabhu, K., Gural, A., Khan, Z.F., Radway, R.M., Giordano, M., Koul, K., Doshi, R., Kustin, J.W., Liu, T., Lopes, G.B., *et al.*: CHIMERA: A 0.92-TOPS, 2.2-TOPS/W Edge AI Accelerator With 2-MByte On-Chip Foundry Resistive RAM for Efficient Training and Inference. IEEE Journal of Solid-State Circuits **57**(4), 1013–1026 (2022)

[123] Ignjatović, D.W. Drago Bailey, Bajić, L.: The Wormhole AI Training Processor. In: 2022 IEEE International Solid-State Circuits Conference-(ISSCC) (2022). IEEE

[124] Vasiljevic, J., Bajic, L., Capalija, D., Sokorac, S., Ignjatovic, D., Bajic, L., Trajkovic, M., Hamer, I., Matosevic, I., Cejkov, A., *et al.*: Compute Substrate for Software 2.0. IEEE Micro **41**(2), 50–55 (2021)

[125] Shrestha, A., Fang, H., Rider, D.P., Mei, Z., Qiu, Q.: In-Hardware Learning of Multilayer Spiking Neural Networks on a Neuromorphic Processor. In: 2021 58th ACM/IEEE Design Automation Conference (DAC), pp. 367–372 (2021). IEEE

[126] Höppner, S., Mayr, C.: SpiNNaker2-Towards Extremely Efficient Digital Neuromorphics and Multi-scale Brain Emulation. Proc. NICE (2018)