CALM: Continual Associative Learning Model via Sparse Distributed Memory

Andrey Nechesov [*,†] and Janne Ruponen [*,†]

Posted Date: 6 November 2025

doi: 10.20944/preprints202511.0430.v1

Keywords: continual learning; associative memory; hybrid architecture; sparse distributed memory; SDM

 doi:10.20944/preprints202511.0430.v1

Article

CALM: Continual Associative Learning Model via Sparse Distributed Memory

Andrey Nechesov † and **Janne Ruponen** *,†

The Artificial Intelligence Research Center of Novosibirsk State University, Novosibirsk, Russia
* Correspondence: ruponez@gmail.com
† These authors contributed equally to this work.

Abstract

Sparse Distributed Memory (SDM) provides a biologically inspired mechanism for associative and online learning. Transformer architectures, despite exceptional inference performance, remain static and vulnerable to catastrophic forgetting. This work introduces Continual Associative Learning Model (CALM), a hybrid framework that integrates SDM with lightweight dual-transformer modules to enable continual, lifelong adaptation without retraining. The architecture maintains an always-online associative memory for episodic storage (System 1) while pair of asynchronous transformer consolidate experience in the background for uninterrupted reasoning and gradual model evolution (System 2). The framework remains compatible with standard transformer benchmarks, establishing a shared evaluation basis for both reasoning accuracy and continual learning stability. Preliminary experiments using the SDMPreMark benchmark evaluate algorithmic behavior across multiple configuration sets, revealing a critical radius-threshold phenomenon in SDM recall and demonstrating near-linear computational scaling with substantial memory efficiency over dense-vector systems. These results represent deterministic characterization of SDM dynamics, establishing empirical baselines for future integration with transformer-based semantic tasks. Planned benchmarks on continual-learning and reasoning datasets (e.g., GLUE, SQuAD, episodic recall) will quantify retention stability, adaptation speed, and long-term reasoning performance. The CALM framework thereby provides a reproducible foundation for studying continual memory and associative learning in hybrid transformer architectures.

Keywords: continual learning; associative memory; hybrid architecture; sparse distributed memory; SDM

1. Introduction

Modern large language models (LLMs) with transformer architecture revolutionized natural language understanding and generation. However, transformers lack persistent memory, continual learning capabilities, and are vulnerable to catastrophic forgetting. The reason for this is the encoding of information through dense vector spaces and positional embeddings. While powerful for static inference, they lack inherent memory and must be retrained or fine-tuned to learn new data. This situation led to the exploration of additional methods to integrate an online learning capabilities and dynamic memory processing to transformers. Since generalization and reasoning are possible only via stored associations, memory is interpretable as a substrate of intelligence. If comparisons to biological brains are used, transformers act as neocortex for reasoning. This is System 2 which needs time and energy and learns slowly to extract structured knowledge. It operates with quick and associative System 1 - functions driven by cortical ensemble of hippocampus, amygdala and basal ganglia.

Transformers apply a non-declarative (implicit) memory logic via probabilistic reconstructions although their behavior simulates declarative knowledge. An actual declarative (explicit) memory is formed from semantic memories (facts) and episodic memories (experiences and events) [1]. Although Retrieval-Augmented Generation (RAG) has been used as a complementary retriever, its lack of

 doi:10.20944/preprints202511.0430.v1

internalization does not satisfy declarative representation, which involves acquisition, consolidation (also online), and long-term retrieval. Episodic memory as a sub-component follows a similar workflow by involving encoding (acquisition), consolidation (storage), and recall (retrieval).

Explorations for declarative augmentation via episodic memory incorporated the study of Kanerva's Sparse Distributed Memory (SDM) framework based on high-dimensional, associative, and distributed memory principles. Kanerva's framework in 1988 [2] established SDM as both a model of human long-term memory and a practical associative memory system. SDM represents data in high-dimensional binary spaces where memory is spread across many storage locations instead of localization. Addresses are compared via Hamming distance, and memory retrieval uses weighted averaging from nearby locations. This approach yields robustness to noise and partial input [2,3].

These principles directly motivate SDM's sparse binary representation and distributed storage mechanisms. The Continual Associative Learning Model (CALM) framework is introduced to fill the gap in the integration of lifelong learning in AI applications. Integrating SDM and a transformer was seen as an opportunity to explore online learning with associative memory in various application spaces. This hybrid network merges SDM module (System 1) that is always online with pair of asynchronous lightweight transformer modules for serving and training (System 2). The serving transformer is active (wake) and replaced periodically by forked shadow transformer performing training and consolidation of information. The research goals were set as follows:

1. *Establish evaluation system to configure optimal parameters for SDM devices*
2. *Validate the behavior of SDM by computational tests from SDM benchmark*
3. *Define and validate preliminary the hybrid architecture containing SDM module (System 1) and dual-transformer module (System 2)*

2. Related Work

Subsequent research has developed sophisticated algorithms for high-dimensional pattern storage and retrieval [4,5]. Subsequent research in 1989 established practical parameters for memory operations [6]. The study of the foundations of SDM reveals how biological neural circuits inspired novel computational approaches, enabling modern applications such as a continual learning, neuromorphic computing, and artificial intelligence (AI) systems. The biological inspiration came from cerebellar cortex models by Marr and Albus, where the cerebellum was proposed as an associative learning device through sparsely distributed neural computations [7,8].

The fundamental difference between transformer and SDM representations lies in their vector encoding. Transformers typically operate on dense floating-point vectors (e.g., 768 or 1024 dimensions in BERT/GPT), requiring substantial compute for each token [9]. A dense floating-point vector may look like the following:

```
[0.12, -0.89, 1.4, 0.03, -0.55, 0.6, 0.0, 1.2]
```

In contrast, SDM leverages sparse binary vectors, often of size $n = 1000$ or more, where each bit is either 0 or 1. These high-dimensional binary vectors facilitate associative recall by proximity (Hamming distance) without the need for exact matches or matrix multiplications. Interference is handled by multiple patterns stored at overlapping locations that form associative links. This characteristic allows memory access in $\mathcal{O}(1)$ time per hard location, significantly reducing the computational load for memory access while providing theoretical capacity that can scale exponentially with dimension for sparse patterns. A sparse binary vector may look like the following:

```
[0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,...] (3/512 bits active)
```

The sparsity enables efficient bitwise operations and dramatically reduces memory requirements for embedded systems. Episodic memory could be achievable as an inherent feature for observer objects such as sensors and IoT devices. Characteristics of SDM highlight the efficiency required from the System 1 running initialization of explicit memory. The correspondence to cerebellar provides

 doi:10.20944/preprints202511.0430.v1

the primary neural inspiration with anatomical mappings from mossy fibers to address inputs, in addition to granule cells to hard locations and parallel fibers to distributed storage connections, and Purkinje cells to output neurons [7,10]. This biological architecture exhibits sparse activation patterns (only small percentages of granule cells active simultaneously) and massive parallelism enabling fast computation, as well as distributed storage with graceful degradation [11,12].

The sparse coding principles of widespread neuroscience research have influenced the SDM development. SDM has been extended through cortical coding models mapping mini- and macro-columnar structures [13] and recent continual-learning implementations [12]. Human hippocampus studies by Rutishauser et al. (2014) found direct evidence for sparse distributed coding in episodic memory, with only small percentages of neurons in hippocampus responding to any given memory target. Each neuron contributes to representations of only a few memories, providing "the most efficient way for neurons in hippocampus to encode episodic memories rapidly. [14]". Cortical column theories have demonstrated connections between SDM principles and cortical organization through research by Rinkus (2010) showing macrocolumns functioning as sparse distributed coding fields. Minicolumns (20 L2/3 pyramidal cells) act as winner-take-all competitive modules enforcing sparseness, with sparse distributed codes consisting of 70 active L2/3 cells from much larger populations [13].

Since then, neuromorphic hardware implementations have achieved remarkable success on the SpiNNaker platform, with Steve Furber's lab incorporating N-of-M rank codes into SDM variants for biologically plausible implementation [15–17]. The SpiNNaker2 chip (2024) supports 153 ARM cores with 19MB on-chip SRAM, dedicated ML and neuromorphic accelerators, and 10x increase in neural simulation capacity per watt [18,19]. Recent work demonstrates the first neuromorphic language model using Event-based Gated Recurrent Units (EGRU), achieving LSTM-equivalent performance with significant energy advantages [20]. In another study, XNOR-Net [21] demonstrated the practical advantages of binary representations, achieving 32× memory savings through binary weight approximations and 58× speedup in convolutional operations by using primarily bitwise operations rather than high-precision floating-point arithmetic. These binary approximations enable state-of-the-art deep networks to run on CPUs in real-time rather than requiring GPU acceleration, making them suitable for portable devices with limited computational resources. The success of XNOR-Net validates the fundamental principle underlying SDM that sparse binary representations can maintain accuracy while providing substantial computational and memory advantages over dense floating-point approaches. Achieved efficiency enables SDM to complement transformer models with declarative memory, but also provides a specialized method to deploy lightweight edge AI with episodic memory capabilities.

Continual learning breakthroughs emerged with Bricken et al.'s 2023 [12] work demonstrating that "Sparse Distributed Memory is a Continual Learner." This modified Multi-Layered Perceptron exhibits strong continual learning capabilities without catastrophic forgetting, translating biological neural circuit principles into artificial networks [12]. The approach achieves continual learning performance comparable to biological systems through sparse activation patterns and biologically-inspired connectivity. In another study of Compressive Sensing SDM (CS-SDM) by Vdovychenko and Tulchinsky [22] integrated a compressive sensing theory with SDM to address classical limitations. This GPU-based implementation uses compressive sensing principles to achieve significantly better capacity and denoising capabilities compared to classical designs, particularly for semantic storage with enhanced noise tolerance [22,23]. In-memory computing implementations reported in Nature Electronics (2020) achieved breakthrough performance using 760,000 phase-change memory devices that perform analog in-memory computing. These systems demonstrate software-equivalent accuracy for language classification, news classification, and hand gesture recognition while providing massive parallel processing with a significant reduction in data movement and energy consumption [4].

Recent work in continual learning has explored Elastic Weight Consolidation (EWC) [24] and replay-based methods [25]. However, these approaches are based on retraining or storing of past data. In contrast, SDM allows for continual learning without parameter adjustment, supporting modular, fault-tolerant memory access.

 doi:10.20944/preprints202511.0430.v1

The biological foundations established several key principles: metabolic efficiency through sparse coding, fault tolerance via the distributed storage, scalability through high-dimensional spaces allowing virtually unlimited memory capacity, computational speed through parallel processing, and generalization through similar representations for similar inputs supporting learning transfer.

Furthermore, sparse coding in neuroscience is related to biological neural networks that should exhibit sparse activation patterns. Only a few percent of neurons are simultaneously active in cortical areas [26–28]. Sparse codes maximize information capacity and energy efficiency [29,30]. Lateral inhibition mechanisms maintain sparsity constraints [29,31]. Population vector decoding enables robust pattern completion [32,33]

3. Materials and Methods

The primary objectives of this study are (1) to characterize the computational dynamics of SDM through the SDMPreMark benchmark, (2) to define a verifiable continual learning loop integrating SDM and transformers, and (3) to establish empirical baselines for System 1 (associative) and System 2 (reasoning) integration. The methodological approach contains five sections. The first section establishes evaluation methodology for SDM operations, according existing literature. The second section establishes SDM benchmark system, which converts SDM evaluation system to algorithmic system running as a computer program to measure SDM capabilities for different devices, aiding with parameter calibration. The third section involves mapping system for System 1 to explore feature integrations from existing biological mechanisms. The fourth section introduces CALM architecture that embeds calibrated SDM module to dual-transformer module. The fifth section studies validation of CALM integration logic.

3.1. SDM Evaluation Methodology

Unlike von Neumann architectures, SDM implements content-addressable memory based on Hamming distance and distributed storage. This section summarizes the key mathematical operations; full derivations follow Kanerva's original formulation [2,11]. **Hamming distance** is defined as $H(x,y)$ and is used for SDM addressing for binary vectors $x,y \in \{0,1\}^n$; in bipolar form $x,y \in \{-1,+1\}^n$, this is simplified, as shown in Equation (1):

$$H(x,y) = \sum_{i=1}^{n} |x_i - y_i| = \sum_{i=1}^{n} (x_i \oplus y_i) \;\rightarrow\; H(x,y) = \frac{n - x \cdot y}{2} \tag{1}$$

where n is the vector dimensionality. Memory locations are activated when $H(A_m, x) \leq r$, or equivalently $A_m \cdot x \geq n - 2r$. **Write operation** is implemented by storing each pattern $\mathbf{w}$ in all active locations within radius r:

$$\mathbf{C} := \mathbf{C} + \mathbf{y} \otimes \mathbf{w}, \tag{2}$$

where $\mathbf{y}$ is a binary activation mask ($y_m = 1$ if $H(A_m, x) \leq r$). Weighted writes use distance-dependent gains $\alpha(H)$ that decrease with radius. **Read operation** retrieves aggregates of all active counters and applies thresholding:

$$\mathbf{z} = \text{sign}(\mathbf{C}^T \mathbf{y}), \tag{3}$$

which reconstructs stored patterns even from incomplete inputs. Retrieval fidelity depends on the signal-to-noise ratio ρ and the resulting bit accuracy ϕ, given by:

$$\rho = \frac{\mu}{\sigma}, \quad \mu = pM, \quad \sigma^2 \approx pM[1 + pT(1 + p^2 M)], \quad \phi \approx \Phi(\rho) \tag{4}$$

where p is the activation probability of a memory location, M is the number of hard locations, T is the number of stored patterns, μ and σ^2 are the signal and noise components respectively, and Φ

 doi:10.20944/preprints202511.0430.v1

denotes the standard normal cumulative distribution function (CDF). **Access radius** is solved by the probability of activating a location p and its optimal value, following Kanerva's rule:

$$p^* = (2MT)^{-1/3}, \quad r \approx 0.4n, \tag{5}$$

where M is the number of hard locations and T the number of stored patterns. The radius controls the trade-off between generalization and interference. The **Address generation** uses uniform random N-bit addresses stored in matrix [16,34]:

$$\mathbf{A} \in \{-1, +1\}^{M \times N} \tag{6}$$

with each row representing one hard location's address in random or structured address matrix. Jaeckel's Selected-Coordinate Design optimizes this through sparse address matrices with $k \approx 10$ non-zero coordinates per address and activation condition:

$$\mathbf{A}_m \cdot \mathbf{x} = k \tag{7}$$

which provide exact match on selected coordinates. The use of Hyperplane applies k ones per hard address for skewed data distributions. **Weighted retrieval** applies similarity-based weighting functions such as

$$w_i(Q) = \exp(-\alpha H(A_i, Q)), \quad \text{for } H(A_i, Q) \leq r \tag{8}$$

and yield smooth recall probabilities:

$$P_{\text{retrieval}}(\text{BER}) = \sum_{k=0}^{r} \binom{n}{k} \text{BER}^k (1 - \text{BER})^{n-k} \tag{9}$$

where $w_i(Q)$ is the weight assigned to location i for query Q, $H(A_i, Q)$ denotes the Hamming distance between the query and memory address A_i, r is the access radius that defines the activation threshold, and α is a decay coefficient controlling distance sensitivity. In the recall probability expression, n is the dimensionality of the binary vector space, k indexes the number of bit errors, and BER (bit-error rate) defines the fraction of corrupted bits in the query relative to the stored address. Further methods involve *location activity probability* $P_{activate}$ and *expected number of activated locations* $E[K]$ (Equation (10)):

$$P_{activate} = P(H(\mathbf{a}_i, \mathbf{p}) \leq r) = \frac{\sum_{k=0}^{r} \binom{d}{k}}{2^d}, \quad E[K] = m \cdot P_{activate} \tag{10}$$

where $H(\mathbf{a}_i, \mathbf{p})$ is the Hamming distance between address $\mathbf{a}_i$ and probe $\mathbf{p}$, m is the total number of memory locations, d is the vector dimensionality, r is the access radius, and k is the summation index over bit differences. *Interference coefficient* γ (patterns per location) and *quality degradation due to interference* $Q(\gamma)$ are defined as (Equation (11)):

$$\gamma = \frac{N_{total}}{E[K]}, \quad Q(\gamma) = \frac{1}{1 + \beta\gamma^2} \tag{11}$$

where N_{total} is the sum of all patterns, β is the interference sensitivity constant that determines how strongly overlapping pattern storage affects recall quality. Higher β values correspond to higher sensitivity and faster degradation of the retrieval accuracy as γ increases. *Shannon-type capacity bound* $C_{Shannon}$ defines the theoretical upper bound on SDM storage capacity per dimension for interference-free storage while *empirical capacity* $C_{empirical}$ measures actual storage efficiency and capacity to store unique patters per location before domination of recall errors (Equation (12)):

$$C_{Shannon} = \frac{2^d \cdot \sum_{k=0}^{r} \binom{d}{k}}{m \cdot d}, \quad C_{empirical} = \frac{N_{succ}}{m} \tag{12}$$

 doi:10.20944/preprints202511.0430.v1

where N_{succ} measures successfully stored patterns. Having established the theoretical foundations for SDM evaluation, next section proceeds to examine how evaluation methodology is converted to benchmark design for real-life performance tests.

3.2. SDM Benchmark Design

LLM benchmarks assume continuous token embeddings, cosine similarity and dot products on continuous vectors, gradient-based learning with continuous optimization, and probabilistic next-token prediction. The combination of memory architecture in terms of address space (learned embeddings versus random binary addresses), storage (parameter matrices vs hard locations with binary counters) and retrieval (attention mechanisms vs Hamming distance-based associative recall) create distinct challenges to apply LLM benchmarks on SDM-based processes.

This study focused on implementing SDMPreMark as the first configuration phase to discover optimal parameters for different processing capabilities. It provides device-specific analysis to evaluate and optimize selection of suitable SDM configuration combinations, fingerprinting optimal parameter combinations (vector_dim, num_locations, access_radius, reinforce, match_ratio) across different hardware environments. This benchmark framework tests 1470 unique configurations, exploring the trade-offs between memory capacity, retrieval accuracy, and computational efficiency. Key parameter ranges include parameter-sweep coverage of vector dimension, memory locations, access radius factor, and reinforcement cycles. The parameters are listed in Table 1.

Table 1. SDM Benchmark Parameter Ranges.

Parameter	**Values**	***Rationale***
Vector Dimension	32, 64, 128, 256, 512, 1024	*Tests scaling from embedded to server deployment*
Memory Locations	500, 1K, 3K, 5K, 8K	*Evaluates capacity vs. interference trade-offs*
Access Radius Factor	0.05, 0.1, 0.2, 0.4, 0.6, 0.78, 0.9	*Explores specificity vs. generalization spectrum*
Reinforcement Cycles	1, 5, 10, 15, 30, 50, 100	*Standard strengthening for reliable storage*

Vector dimension impacts memory footprint and computational complexity. Larger dimensions establish more precise representations but require proportionally more resources. Access radius is calculated as $r = \max(1, \lfloor d \times \alpha \rfloor)$ where d is vector dimension and α is the radius factor, controlling the specificity-generalization issues in pattern matching. Memory locations determines storage capacity and potential interference patterns, where higher counts provide greater capacity at the cost of increased search complexity. Reinforcement cycles control pattern strengthening, with more cycles improving storage reliability but increasing training overhead.

The benchmark generates key performance indicators including match ratio (retrieval accuracy ranging 0-1), execution duration (computational efficiency in seconds), and sparsity ratio (input pattern density). Additional metrics capture input pattern characteristics (ones count), recalled pattern fidelity (recalled ones count), and derived measures such as the radius factor for systematic analysis. These metrics enable comprehensive evaluation of configuration performance across accuracy, efficiency, and pattern handling dimensions. The benchmark aims to identify parameter combinations that achieve high retrieval accuracy with minimal computational overhead for specific hardware environments.

Further optimization of SDM could involve a benchmark for evaluating episodic memory capacity, pattern completion accuracy, and interference resistance in long-context scenarios using binary distributed representations. Another benchmark design could be used for assessing temporal reasoning capabilities through binary vector binding/unbinding operations and chronological sequence reconstruction in high-dimensional space.

 doi:10.20944/preprints202511.0430.v1

3.3. System 1 Methodology

According to Kahneman [35], System 1 in biological context involves plethora of mapped mechanisms applied for fast thinking. The most prevalent of those is associative activation, the core feature of SDM method. SDM also achieves stereotyping as category vectors naturally serve as prototypes via pattern matching. SDM also integrates mechanisms for availability heuristic as retrieval probability scales with memory frequency and recency. Similarity-based recall functions provides classification by resemblance and prototype matching, emerging as representativeness heuristic. Since SDM limit local view, it models partial-context reasoning, resembling the mechanism of "What You See Is All There is" (WYSIATI).

Questions were raised as to whether other mechanisms could be beneficial and implementable for CALM. The easiest mechanisms for integration were estimated to be substitution, anchoring, cognitive ease, halo effect, and framing effect.

Substitution simplifies difficult questions and favors intuition. Its use in the SDM framework adds fallback mechanism under uncertainty with incomplete data. Substitution requires addition of heuristic controller to fall back to nearest simpler query under high uncertainty. Risk of oversimplification (confident answers) is apparent, and therefore the mechanism should be used as probabilistic feedback to be triggered only in low confidence levels, marked as heuristic inference.

Anchoring provides stability and context persistence across iterative queries, and prevents erratic jumps in associative recall. Anchoring could operate by maintaining an initial retrieval with decay per distance or time. The risk of path fixation may occur during early states, requiring controller decay by using System 2 to reset anchor periodically.

Cognitive ease in SDM is the confidence estimator which signals for internal coherence by fast retrieval and low-conflict responses. Retrieval fluency drives confidence signal for cognitive ease. The overrating of effortless associations may ignore deeper associations. Consequently, this mechanism should be used to weight certainty estimates only, and not for logical validity or decision rule.

Halo effect (influence of overall impression to generalize positively all traits) may be used to propagate affective coherence such as consistent emotional stance or tone across related memory clusters. In practice, this could manifest as empathy, narrative consistency, and user alignment. This mechanism applies spread valence to nearby vectors for tone propagation. The risk of halo effect is the spread of undesired bias to overly positive or negative framing. Therefore, it should be applied only locally and allow the decay over distance. An additional module may be required to neutralize overgeneralization.

For stylistic purposes, framing effect can bring more sensitivity to linguistic context, helping the system to detect user framing adapt tone based on keyword frames. Acting as a stylistic modulation for the encoder, it could provide an additional record of framing tags separate from factual memory.

Mechanisms deemed not worth of feature integration include confirmation bias, the law of small numbers, and overconfidence. Confirmation bias is problematic due to its rejection of new or contradictory data, introducing riks of rigidity and echo-chamber effect. These are not interpreted as desirable features for adaptive, learning model. A law of small numbers (generalization based on small sample sizes) is neither desirable behavior; it may however manifest itself at the degree when adaptive memory radius is tuned. Likewise, overconfidence can occur during natural manifestation of SDM parameters. Consequently, these methods act more as emergent lag indicators of underlying SDM parameters, instead of driving mechanisms for memory processing.

Additionally, while intensity matching, affect heuristic, and causal thinking were estimated to be not integrated to System 1, although applicable when integrated with System 2. Intensity matching would enable along reasoning about degree and magnitude to form analogues, emotional scales and prioritization, but requires scalar extensions to binary SDM. Affect heuristic is critical for motivational steering, empathy, and adaptivity, but is difficult to ground safely without human-level affection. Furthermore, although causal thinking with predictive reasoning is useful for planning and behavior modeling, its implementation requires temporal reasoning beyond SDM module.

 doi:10.20944/preprints202511.0430.v1

Mechanisms of System 1 in the context of SDM are represented in four categories: inherent, intuitive, emergent, and causal. Mirroring cognitive architectures, the inherent module act as reflexive neural wiring; the intuitive module applies heuristic reasoning; the emergent module displays adaptive biases; and the causal module is reflective cognition with abstractions (System 2 interface). This also clarifies evolutionary design priorities:

1. Establish inherent System 1 mechanisms
2. Integrate intuitive System 1 mechanisms
3. Implement controller to detect emergent System 1 mechanisms
4. Integrate System 2 dual transformers to System 1 (SDM and controller)

3.4. CALM Architecture

Building on limitations of conventional transformer systems and the continual learning capabilities of Sparse Distributed Memory (SDM), the proposed CALM framework integrates a dual-system architecture inspired by cognitive processes (Figure 1). This hybrid design was motivated by three key goals:

- **Continual Adaptation**: SDM is allowing rapid incorporation of new data without retraining.
- **Memory Grounding**: The system maintains an episodic trace of events, enabling structured recall and System 1 operations.
- **Biological Plausibility**: The modular memory structure aligns with the distributed intelligence of the cortical columns and sensorimotor integration.

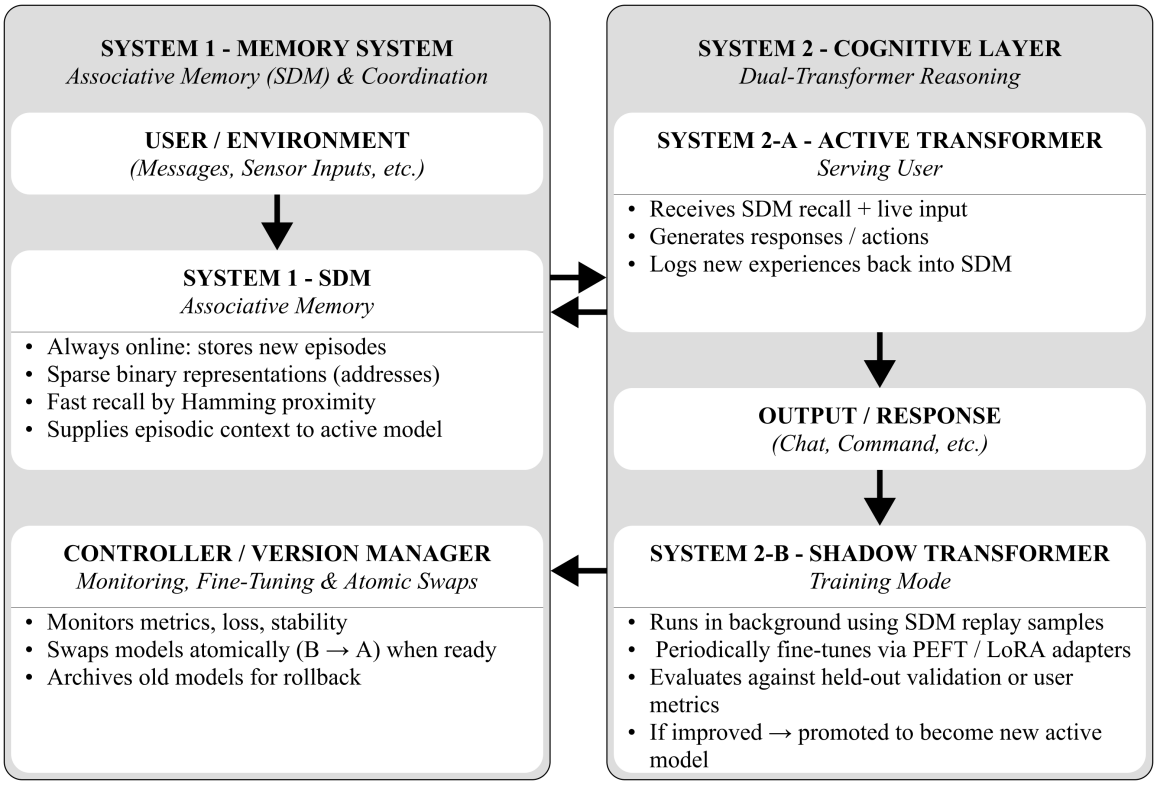

Figure 1. Hybrid system combining transformer encoders with SDM storage. The pipeline processes sensory input through four stages: feature extraction, binary encoding, associative storage, and decision making.

System 1 encapsulates online associative memory (SDM), storing new experiences as sparse binary vectors, and providing rapid associative recall via Hamming proximity. It also provides episodic context for active models. System 2 comprises an *Active Transformer* (System 2-A) serving user interactions and a *Shadow Transformer* (System 2-B) training asynchronously using SDM replay samples and fine-tuning adapters (PEFT/LoRA). The shadow model replaces the active one atomically, when

 doi:10.20944/preprints202511.0430.v1

validated improvement occurs. The controller and version manager monitor metrics, stability, and loss, but also manage swaps and archives model versions for rollback or analysis.The model operates in three concurrent processes:

1. **Encoding & Acquisition:** New input is encoded as high-dimensional binary vectors and stored in SDM.
2. **Retrieval & Reasoning:** SDM provides associative recall to the Active Transformer, which generates responses and updates memory into the SDM.
3. **Consolidation:** The Shadow Transformer fine-tunes asynchronously on SDM replay data, ensuring gradual learning without disrupting live performance.

This architecture enables continual adaptation, episodic recall, and stable long-term reasoning—bridging the gap between static transformer inference and dynamic memory-based cognition. Internally, CALM's distributed SDM implementation mirrors cortical organization through modular associative units analogous to cortical columns. In *local learning*, each SDM module independently encodes local sensory or contextual input, enabling online adaptation without global retraining. The use of *reference frames* are implemented via spatial and temporal patterns that are represented as high-dimensional binary vectors, preserving relative relationships across inputs. *Voting mechanism* uses overlapping modules collaborating through similarity-based activation to resolve ambiguous or noisy inputs. Higher-level SDM layers integrate outputs from lower layers to form increasingly abstract episodic representations, representing *hierarchical integration*.

Transformer-to-binary encoding serves as a bridge between dense transformer embeddings and sparse SDM representations. Floating-point vectors are constrained by threshold to produce binary encoding, ensuring compatibility between systems while preserving semantic content:

$$\mathbf{v}_{dense} = \text{BERT}(\text{tokenize}(s)) \in \mathbb{R}^{d_{model}} \tag{13}$$

$$b_i = \begin{cases} 1, & \text{if } v_i > \theta \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

Here, *BERT* denotes a bidirectional transformer encoder that maps a tokenized input sequence s into a dense embedding space of dimensionality d_{model}. The vector $\mathbf{v}_{dense}$ represents the continuous embedding output of the transformer, b_i is the i-th bit of the resulting binary vector after thresholding, and θ is the activation threshold controlling sparsity. Alternative mapping schemes such as locality-sensitive hashing (LSH) or learned quantization may be applied to balance sparsity and information retention.

The SDM maintains a fixed set of hard memory locations uniformly distributed in high-dimensional space. Stored patterns activate subsets of these locations within a Hamming radius r. Fixed locations simplify hardware access, though future work may explore dynamic expansion for scalability. An example of SDM association storage between high-dimensional addresses and their corresponding data is demonstrated in (Table 2).

Table 2. Example SDM Memory Contents.

Address Vector (Binary)	Payload	Confidence
001010001000010100...	"person detected"	0.87
111000000101100010...	"car detected"	0.92
000100100000000001...	"no motion"	0.95

Memory retrieval operates by computing Hamming distances between query and stored addresses and aggregating all locations within a threshold. SDM tolerates noisy or partial inputs through distributed representation and weighted recall, recovering nearest stored patterns even under corruption (Equation (15)).

 doi:10.20944/preprints202511.0430.v1

$$\text{BER} = \frac{\text{corrupted\_bits}}{d}, \quad P_{retrieval}(\text{BER}) = \sum_{k=0}^{r} \binom{d}{k} \text{BER}^k (1 - \text{BER})^{d-k}. \tag{15}$$

where d denotes the dimensionality of the binary address vectors, r is the Hamming distance threshold (access radius), and k indexes the number of bit errors tolerated during retrieval. $P_{retrieval}$ thus expresses the probability of successful recall under a Bernoulli noise model with bit error rate BER.

SDM tolerates noisy or partial inputs due to its distributed representation and weighted retrieval. For instance, a corrupted binary query averages activations over nearby locations, recovering the closest stored pattern. This contrasts with transformers that degrade under corrupted inputs unless retrained. Experimental results demonstrate this behavior (Section 4).

The CALM architecture unites the reasoning power of transformers with the continual memory and robustness of SDM. System 1 provides online associative learning and recall, while System 2 performs symbolic reasoning and long-term consolidation through dual transformer models. This design supports continual adaptation, structured episodic recall, and biologically plausible memory dynamics. Architectural trade-offs for the design are summarized in Table 3.

Table 3. SDM vs. Transformer Trade-offs generalized.

Capability	SDM Strength	Transformer Strength
Online Learning	High	Low
Memory Efficiency	High	Low
Language Modeling	Low	High
Few-shot Learning	High	Medium
Noise Robustness	High	Medium
Complex Reasoning	Medium	High
Interpretability	High	Low

This hybrid approach is particularly suited for systems operating in dynamic or noisy environments, such as embedded agents and real-time anomaly detection tasks, where continual learning and long-term memory stability are critical.

In the following section, the experimental setup is defined for its use to validate this architecture, including integration with embedded sensors and benchmarks against conventional transformer models.

3.5. CALM Integration

The CALM framework integrates Sparse Distributed Memory (System 1) with dual-transformer reasoning modules (System 2-A and 2-B) to enable continual, online adaptation. While the SDMPreMark benchmark empirically validates the stability and efficiency of the System 1 memory substrate, the overall CALM architecture must also demonstrate provable capability for lifelong continual learning by satisfying three conditions. **(1) Theoretical Sufficiency.** The CALM architecture satisfies the three canonical operations required for continual learning:

$$\mathcal{E}(x_t) \xrightarrow{\text{encode}} \text{SDM}, \quad \mathcal{R}(C_t) \xrightarrow{\text{recall}} \text{Transformer}, \quad \mathcal{U}(\hat{x}_t) \xrightarrow{\text{consolidate}} \text{SDM} \tag{16}$$

where $\mathcal{E}$ denotes episodic encoding, $\mathcal{R}$ associative retrieval providing contextual recall to the active transformer, and $\mathcal{U}$ consolidation of newly generated representations back into memory. Together, these three transformations form a closed-loop continual learning cycle:

$$(\text{Encode} \rightarrow \text{Recall} \rightarrow \text{Consolidate})_t \Rightarrow (\text{Adaptive Memory Update})_{t+1}$$

This cycle ensures that every interaction (user input, environmental stimulus, or transformer output) is internalized into SDM and available for subsequent reasoning.

 doi:10.20944/preprints202511.0430.v1

(2) Empirical Plausibility. Results from the SDMPreMark benchmark establish a deterministic *critical radius threshold* ($r \approx 0.4$–0.6 of vector dimensionality) defining stable associative recall. Above this threshold, SDM recall becomes error-free (match_ratio = 1.0) across all tested dimensions, satisfying the stability condition required for consistent transformer context generation. This empirical property guarantees that the memory subsystem feeding the reasoning layers operates deterministically and noise-tolerantly.

(3) Provability Condition. Functional provability of CALM is defined through measurable improvement of the active transformer following SDM-based consolidation. Given the shadow transformer B_t trained asynchronously on replay samples $\mathcal{M}_{t-r:t}$ from SDM memory, promotion to active model A_{t+1} occurs when:

$$\Delta\mathcal{L}_t = \mathcal{L}(A_t) - \mathcal{L}(B_t) > \epsilon \tag{17}$$

where $\mathcal{L}$ is a task-specific loss (e.g., validation loss, accuracy, or user metric) and ϵ is a minimal improvement threshold. This measurable criterion provides a falsifiable condition for continual learning: each update cycle must yield a quantifiable improvement in the active model's performance without external retraining [24].

(4) Verification Roadmap. The provability of CALM can be established through a progressive evaluation sequence. At first, *SDMPreMark* benchmark confirms associative stability and computational efficiency (System 1 validation). Next verification component, *CALMark* benchmark measures $\Delta\mathcal{L}$ before and after atomic model swaps to verify memory-driven consolidation (System-2 validation). Third verification component involves *continual reasoning* tests employing long-term retention and adaptation benchmarks (e.g., GLUE, SQuAD) to validate stability over time. Verification roadmap is categorized in Table 4.

Table 4. Verification stages of CALM framework.

Stage	Input	Output Metrics	Objective
SDMPreMark	SDM configuration sets	Match ratio, sparsity, runtime	Validate System 1 associative stability
CALMark	SDM replay data, transformer checkpoints	$\Delta\mathcal{L}$, loss stability, swap success	Verify continual consolidation between System 1 and System 2
Continual Reasoning	GLUE, SQuAD, episodic datasets	Retention, adaptation speed, reasoning accuracy	Measure long-term stability and learning efficiency

4. Preliminary Results

The SDMPreMark benchmark was executed as a controlled in silico experiment to map parameter sensitivities in high-dimensional associative memory operations. Each configuration was deterministically evaluated via the the sdm memory test routine, providing reproducible metrics (match ratio, sparsity ratio, runtime). Although these experiments do not involve learned datasets, they qualify as algorithmic characterization of SDM dynamics.

Critical Radius Threshold Behavior: The most striking finding is the existence of a sharp performance threshold around radius factor 0.2 (corresponding to access radius values of 6, 12, 25, 51, 102, 204, 307, 399, 460, 614, 798, and 921 for vector dimensions 32, 64, 128, 256, 512, and 1024 respectively). This transition corresponds to the empirical critical radius function $r_{critical}$ (Equation (5)) and performance transition function $P_{success}$ (Equation (10)), presented together in Equation (18).

$$r_{critical}(d) = \begin{cases} 0.375d, & \text{if } d \leq 64 \\ 0.40d, & \text{if } 64 < d \leq 256, \\ 0.60d, & \text{if } d > 256 \end{cases} \quad P_{success}(r,d) = \begin{cases} 0.4 - 0.6, & \text{if } r < r_{critical}(d) \\ 1.0, & \text{if } r \geq r_{critical}(d) \end{cases} \tag{18}$$

 doi:10.20944/preprints202511.0430.v1

Below this threshold, match ratios consistently remain under 0.7, while configurations at or above radius factor 0.2 achieve perfect recall (match_ratio = 1.0) across most parameter combinations. This threshold represents a fundamental transition point where the SDM system shifts from under-activation to reliable pattern retrieval.

Vector Dimension Scaling: Performance remains remarkably stable across vector dimensions from 32 to 1024 bits once the critical radius threshold is exceeded. Lower dimensions (32-128) show slightly more variability in the sub-threshold region, while higher dimensions (512-1024) demonstrate more consistent behavior. Notably, the M1 Pro's unified memory architecture appears to handle even the largest vector dimensions efficiently, with execution times scaling predictably rather than experiencing memory bottlenecks.

Computational Efficiency Patterns: Empirical latency model $T_{empirical}$ was derived from time complexity equations to extend theoretical latency scaling). Execution duration scales linearly with both memory locations and reinforcement cycles ($R^2 = 0.98$). The relationship follows: $T \approx 0.07 \times \frac{\text{locations}}{1000} + 0.03$ milliseconds. The binary nature of retrieval success patterns creates distinct operating regimes, as show in Table 5. Latency was calculated according to the empirical relation (Equation (19)):

$$T_{empirical}(d, m) = 0.07 \cdot \frac{m}{1000} + 0.03 \text{ ms}, \quad E_{bit} = \alpha_{switching} \cdot V_{dd}^2 + \beta_{leakage} \cdot V_{dd} \tag{19}$$

Table 5. SDM Operating Regimes by Access Radius.

Regime	Radius Range	Match Ratio	Characteristics
Under-activation	$r < 0.4d$	0.4–0.6	No successful retrieval, recalled_ones = 0
Transition Zone	$r \approx 0.4d$	0.6–0.9	Unstable, configuration-dependent
Over-activation	$r > 0.4d$	1.0	Perfect recall, full pattern recovery

Configurations with 500-1000 memory locations complete in under 0.1 seconds for most parameter combinations, while 8000 locations require 1-5 seconds. The reinforcement cycle impact is particularly pronounced, with 100 cycles taking approximately 10x longer than single-cycle operations. This suggests that for the M1 Pro platform, configurations with 1000-3000 memory locations and 10-30 reinforcement cycles provide an optimal balance between capacity and responsiveness.

Optimal Radius Selection: To minimize retrieval errors, the optimal radius r^* is obtained by balancing miss and interference probabilities: Equation (20) contains methods for optimal radius selection r^* and miss probability $P_{miss}(r)$ (derived from Equation (10)).

$$r^* = \arg\min_r \left[\alpha \cdot P_{miss}(r) + \beta \cdot P_{interference}(r)\right], \quad P_{miss}(r) = P(_{idle}) = \left(1 - \frac{\sum_{k=0}^{r} \binom{d}{k}}{2^d}\right)^m \tag{20}$$

Capacity: Following Equation (12), practical efficiency was calculated with capacity utilization factor $\alpha_{capacity}$ (Equation (21)). Increasing memory locations shows minimal effect on match ratio but significant impact on processing time (Table 6).

$$\alpha_{capacity} = \frac{C_{empirical}}{C_{Shannon}} \approx 0.3 - 0.7 \tag{21}$$

 doi:10.20944/preprints202511.0430.v1

Table 6. Memory Scaling Effects (1024-bit vectors, radius = 614).

Locations	Match Ratio	Latency (ms)	Efficiency
500	1.0	0.090	High
1,000	1.0	0.188	Medium
3,000	1.0	0.564	Medium
5,000	1.0	0.942	Low
8,000	1.0	1.501	Low

The M1 Pro's performance characteristics reveal that this platform favors moderate memory locations (1000-5000) with radius factors between 0.4-0.6 for optimal efficiency. Configurations exceeding these parameters show diminishing returns in accuracy, while significantly increasing computational overhead. These findings establish a clear device-specific configuration profile that prioritizes the M1 Pro's strengths in parallel processing while respecting its thermal and power constraints.

Analytical Resource Constraints: To quantify the stability of associative recall under concurrent storage, the interference probability $P_{interference}$ (Equation (22)) was derived from the expected number of activated locations $E[K]$. This metric captures how overlapping address activations degrade recall fidelity as memory utilization increases.

$$P_{interference}(r) = 1 - \exp\left(-\frac{N \cdot E[K]}{m}\right) \tag{22}$$

Memory requirements M_{total} and M_{memory} (Equation (23)) were evaluated to estimate storage scalability (Equation (12):

$$M_{total} = m \cdot (d_{addr} + d_{data} + \log_2(N_{max})), \quad M_{memory}(d, m) = m \cdot (d + p) \tag{23}$$

Equation (24) dictates processing power (operations per second) P_{ops} and time complexities (T_{write}, T_{read}) according to Equation (1):

$$P_{ops} = \frac{f_{cpu}}{d \cdot m}, \quad T_{write}(d, m) = O(m \cdot d) T_{read}(d, m, k) = O(m \cdot d + k \cdot \log k) \tag{24}$$

These formulations were used only to validate scalability trends observed empirically and were not directly benchmarked on hardware.

5. Discussion and Future Work

The results demonstrate SDM's viability for real-time applications while highlighting the critical importance of proper sparsity control in the encoding pipeline. Recommendations for deploying SDM-based AI for different hardware setups include three recommendations. Firstly, sparsity problem arises from 50% bit activation which is too dense for efficient SDM operation, requiring larger radii and reducing specificity. *Sparse encoding* should be therefore implemented to encode schemes targeting 2-5% sparsity instead of 50%. Secondly, sharp transition between failure and success suggests sensitivity to parameter tuning based on access radius, contradicting assumptions of gradual degradation. Critical radius seems to grow faster than linear with dimension, challenging traditional SDM assumptions Thus, achieving sufficient binary performance requires *adaptive radius* that can be achieved by dynamic radius adjustment based on vector dimension and content. Thirdly, *capacity optimization* for embedded deployment with 500-1000 locations provides optimal trade-off for latency-accuracy.

Successful deployment of the CALM model could allow AI agents, Internet-of-Things (IoT) devices and cognitive robotics the lifelong learning and memory-augmented decision-making with high energy efficiency. One promising direction for extending CALM involves applying its associative memory and continual-learning capabilities to collective decision-making systems. Recent work in civic intelligence and smart-city governance is suggesting that integrating blockchain and AI can enhance participatory processes through transparent data flows and adaptive governance models [36]. CALM or similar frameworks could serve as the cognitive substrate for such civic intelligence

 doi:10.20944/preprints202511.0430.v1

frameworks by enabling persistent, decentralized memory and reasoning across distributed agents. This would allow communities and institutions to develop continual learning infrastructures for governance.

Existing smart device assistants respond to isolated prompts. In contrast, continuity of purpose in CALM handles prompts as storage of declarative knowledge. Progress mirrors, contextual nudges, and surface memory associations could create a meta-cognitive AI assistant set to grow with user's goals. A next step is to project persistent goals in digital societies toward realizing autonomous, self-evolving Virtual Cities (VCs) and its smart citizens, integrated into continual adaptive learning architectures. By introducing sparse distributed memory consolidation, online updating, and self-referential feedback loops, AI entities mirror their goals and cognitive models in real time from context providers. Within the context of VCs, quantifying autonomy levels in AI entities and agents is challenging as they should transcend from simple digital replicas of real-world systems, and then evolve gradually toward more complex mirroring in socio-technical ecosystems [37]. Building on continual associative architecture, future research could extend the framework into large-scale digital twin ecosystems representing adaptive VCs. Agents within a hypernetwork would possess local SDM and reasoning capabilities learning from real-world IoT data and interactions while sharing cognitive substrate between each other [38].

An open direction is to explore distributed polynomial-time SDM and AI updates via the block-wise memory allocation, which could resemble polynomial proof-of-memory mechanisms from blockchain theory [39]. This would allow decentralized and scalable lifelong learning in edge environments. The stake of resources proven in this context involves a scalar measure of cognitive bandwidth for verifiable memory traces. Participants in such a network could exchange verified experience blocks, where each block encapsulates episodic data, update metadata, and a cryptographic certificate of associative gain. This could allow decentralized agents to share verified cognitive resources instead of raw computation, establishing an economy of useful learning in which value is derived from validated memory formation and recall performance. The distributed SDM network introduces a new staking resource method for decentralized economics, which presents an alternative to existing resource methods such as computation resources or energy expenditure.

6. Conclusion

Sparse Distributed Memory offers a promising path toward more adaptive, efficient, and human-like AI systems. By embracing distributed, continual memory, and biologically inspired computation, interactions with AI can surpass the static nature of current models. Empirical analysis reveals a critical radius threshold ($r \approx 0.4$–0.6) that defines stable associative recall and demonstrates near-linear computational scaling. The SDMPreMark benchmark introduced provided an initial phase to codify the evaluation methodology, allowing the discovery of optimal parameters per host. This confirms the associative stability and computational efficiency required from System 1.

The architecture for a hybrid LLM scheme with episodic memory capacities was introduced, containing System 1 (SDM) and System 2-A/B (dual-transformers). An evolutionary design of System 1 was introduced to map feature clusters for further behavioral optimizations. The next step involves adding intuitive models to SDM, and integrating lightweight transformer pair and synchronization of operations with existing SDM modules applied in SDMPreMark. The final deployment also requires that emergent mechanisms are trackable and causal linkages between System 1 and System 2 are established. An additional benchmark is planned as the method to verify memory-driven consolidations before and after atomic models, along with continual reasoning tests (e.g, GLUE, SQuAD) to validate stability over time.

Although SDMPreMark provides a deterministic evaluation, the absence of semantic data benchmarks limits the generalization assessment. Future CALMark experiments will address this by incorporating transformer-driven semantic replay.

 doi:10.20944/preprints202511.0430.v1

The implication of the main findings establishes a biologically grounded framework for lifelong AI learning that maintains compatibility with standard transformer benchmarks and adjusts to different device configurations.

Author Contributions: For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used "Conceptualization, A.N. and J.R.; methodology, J.R.; software, J.R.; validation, A.N.; formal analysis, A.N.; investigation, A.N. and J.R.; resources, J.R.; data curation, J.R.; writing—original draft preparation, A.N. and J.R; writing—review and editing, A.N. and J.R.; visualization, J.R.; supervision, A.N.; project administration, A.N.; funding acquisition, A.N. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a grant for research centers, provided by the Ministry of Economic Development of the Russian Federation in accordance with the subsidy agreement with the Novosibirsk State University dated April 17, 2025 No. 139-15-2025-006: IGK 000000Ts313925P3S0002.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Benchmark implementation and example scripts will be made available at https://github.com/toxom/CALM upon publication.

Acknowledgments: Authors would like to express gratitude to the Artificial Intelligence Research Center of Novosibirsk State University for their support of this research.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
CALM	Continual Associative Learning Model
EGRU	Event-based Gated Recurrent Units
IoT	Internet of Things
LLM	Large Language Model
RAG	Retrieval-Augmented Generation
SDM	Sparse Distributed Memory
VC	Virtual Cities
WYSIATI	What You See Is All There Is

References

1. Huang, C.C.; Rolls, E.T.; Hsu, C.C.H.; Feng, J.; Lin, C.P. Extensive Cortical Connectivity of the Human Hippocampal Memory System: Beyond the "What" and "Where" Dual Stream Model. *Cerebral Cortex* **2021**, *31*, 4652–4669. https://doi.org/10.1093/cercor/bhab113.
2. Kanerva, P. *Sparse Distributed Memory*; MIT Press: Cambridge, MA, USA, 1988.
3. Kanerva, P. Self-propagating search: a unified theory of memory (address decoding, cerebellum). 1984.
4. Karunaratne, G.; Le Gallo, M.; Cherubini, G.; Benini, L.; Rahimi, A.; Sebastian, A. In-memory hyperdimensional computing. *Nature Electronics* **2020**, *3*, 327–337. Publisher: Nature Publishing Group, https://doi.org/10.1038/s41928-020-0410-3.
5. Keeler, J.D. Capacity for patterns and sequences in Kanerva's SDM as compared to other associative memory models, Denver, CO, 1988. NTRS Author Affiliations: NASA Ames Research Center NTRS Document ID: 19890041660 NTRS Research Center: Legacy CDMS (CDMS).
6. Flynn, M.J.; Kanerva, P.; Bhadkamkar, N. *Sparse distributed memory: Principles and operation*; Number RIACS-TR-89-53, NASA, 1989. NTRS Author Affiliations: Research Inst. for Advanced Computer Science NTRS Document ID: 19920001076 NTRS Research Center: Legacy CDMS (CDMS).

 doi:10.20944/preprints202511.0430.v1

7. Marr, D. A theory of cerebellar cortex. *The Journal of Physiology* **1969**, *202*, 437–470. https://doi.org/10.1113/jphysiol.1969.sp008820.
8. Albus, J.S. Mechanisms of planning and problem solving in the brain. *Mathematical Biosciences* **1979**, *45*, 247–293. https://doi.org/10.1016/0025-5564(79)90063-4.
9. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019, [arXiv:cs.CL/1810.04805].
10. Kawato, M.; Ohmae, S.; Hoang, H.; Sanger, T. 50 Years Since the Marr, Ito, and Albus Models of the Cerebellum. *Neuroscience* **2021**, *462*, 151–174. https://doi.org/10.1016/j.neuroscience.2020.06.019.
11. Kanerva, P. Sparse distributed memory and related models. Technical Report NASA-CR-190553, 1992. NTRS Author Affiliations: Research Inst. for Advanced Computer Science NTRS Document ID: 19920021480 NTRS Research Center: Legacy CDMS (CDMS).
12. Bricken, T.; Davies, X.; Singh, D.; Krotov, D.; Kreiman, G. Sparse Distributed Memory is a Continual Learner. 2023. https://doi.org/10.48550/arXiv.2303.11934.
13. Rinkus, G.J. A cortical sparse distributed coding model linking mini- and macrocolumn-scale functionality. *Frontiers in Neuroanatomy* **2010**, *4*. Publisher: Frontiers, https://doi.org/10.3389/fnana.2010.00017.
14. Wixted, J.T.; Squire, L.R.; Jang, Y.; Papesh, M.H.; Goldinger, S.D.; Kuhn, J.R.; Smith, K.A.; Treiman, D.M.; Steinmetz, P.N. Sparse and distributed coding of episodic memory in neurons of the human hippocampus. *Proceedings of the National Academy of Sciences* **2014**, *111*, 9621–9626. Publisher: Proceedings of the National Academy of Sciences, https://doi.org/10.1073/pnas.1408365111.
15. Snaider, J.; Franklin, S.; Strain, S.; George, E.O. Integer sparse distributed memory: Analysis and results. *Neural Networks* **2013**, *46*, 144–153. https://doi.org/10.1016/j.neunet.2013.05.005.
16. Furber, S.B.; John Bainbridge, W.; Mike Cumpstey, J.; Temple, S. Sparse distributed memory using *N*-of-*M* codes. *Neural Networks* **2004**, *17*, 1437–1451. https://doi.org/10.1016/j.neunet.2004.07.003.
17. Peres, L.; Rhodes, O. Parallelization of Neural Processing on Neuromorphic Hardware. *Frontiers in Neuroscience* **2022**, *16*, 867027. https://doi.org/10.3389/fnins.2022.867027.
18. Gonzalez, H.A.; Huang, J.; Kelber, F.; Nazeer, K.K.; Langer, T.; Liu, C.; Lohrmann, M.; Rostami, A.; Schöne, M.; Vogginger, B.; et al. SpiNNaker2: A Large-Scale Neuromorphic System for Event-Based and Asynchronous Machine Learning **2024**. arXiv:2401.04491 [cs], https://doi.org/10.48550/arXiv.2401.04491.
19. Liu, C.; Bellec, G.; Vogginger, B.; Kappel, D.; Partzsch, J.; Neumärker, F.; Höppner, S.; Maass, W.; Furber, S.B.; Legenstein, R.; et al. Memory-Efficient Deep Learning on a SpiNNaker 2 Prototype. *Frontiers in Neuroscience* **2018**, *12*. Publisher: Frontiers, https://doi.org/10.3389/fnins.2018.00840.
20. Nazeer, K.K.; Schöne, M.; Mukherji, R.; Mayr, C.; Kappel, D.; Subramoney, A. Language Modeling on a SpiNNaker 2 Neuromorphic Chip, 2023. arXiv:2312.09084 [cs] version: 1, https://doi.org/10.48550/arXiv.2312.09084.
21. Rastegari, M.; Ordonez, V.; Redmon, J.; Farhadi, A. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks, 2016, [arXiv:cs.CV/1603.05279]. https://doi.org/10.48550/arXiv.1603.05279.
22. Vdovychenko, R.; Tulchinsky, V. Sparse Distributed Memory for Sparse Distributed Data. In Proceedings of the Intelligent Systems and Applications; Arai, K., Ed., Cham, 2023; p. 74–81. https://doi.org/10.1007/978-3-031-16072-1_5.
23. Vdovychenko, R.; Tulchinsky, V. Sparse Distributed Memory for Binary Sparse Distributed Representations. In Proceedings of the Proceedings of the 2022 7th International Conference on Machine Learning Technologies, New York, NY, USA, 2022; ICMLT '22, p. 266–270. https://doi.org/10.1145/3529399.3529441.
24. 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* **2017**, *114*, 3521–3526. arXiv:1612.00796 [cs], https://doi.org/10.48550/arXiv.1612.00796.
25. Rolnick, D.; Ahuja, A.; Schwarz, J.; Lillicrap, T.P.; Wayne, G. Experience Replay for Continual Learning, 2019. arXiv:1811.11682 [cs], https://doi.org/10.48550/arXiv.1811.11682.
26. Graham, D.; Field, D. Sparse Coding in the Neocortex. *Evolution of Nervous Systems* **2007**, *3*. https://doi.org/10.1016/B0-12-370878-8/00064-1.
27. Beyeler, M.; Rounds, E.L.; Carlson, K.D.; Dutt, N.; Krichmar, J.L. Neural correlates of sparse coding and dimensionality reduction. *PLoS Computational Biology* **2019**, *15*, e1006908. https://doi.org/10.1371/journal.pcbi.1006908.

 doi:10.20944/preprints202511.0430.v1

28. Jääskeläinen, I.P.; Glerean, E.; Klucharev, V.; Shestakova, A.; Ahveninen, J. Do sparse brain activity patterns underlie human cognition? *NeuroImage* **2022**, *263*, 119633. https://doi.org/10.1016/j.neuroimage.2022.119633.
29. Drix, D.; Hafner, V.V.; Schmuker, M. Sparse coding with a somato-dendritic rule. *Neural Networks* **2020**, *131*, 37–49. https://doi.org/10.1016/j.neunet.2020.06.007.
30. Olshausen, B.A.; Field, D.J. Sparse coding of sensory inputs. *Current Opinion in Neurobiology* **2004**. https://doi.org/10.1016/j.conb.2004.07.007.
31. Le, N.D.H. Sparse Code Formation with Linear Inhibition **2015**. arXiv:1503.04115 [cs], https://doi.org/10.48550/arXiv.1503.04115.
32. Panzeri, S.; Moroni, M.; Safaai, H.; Harvey, C.D. The structures and functions of correlations in neural population codes. *Nature Reviews Neuroscience* **2022**, *23*, 551–567. https://doi.org/10.1038/s41583-022-00606-4.
33. Chaisanguanthum, K.S.; Lisberger, S.G. A Neurally Efficient Implementation of Sensory Population Decoding. *The Journal of Neuroscience* **2011**, *31*, 4868–4877. https://doi.org/10.1523/JNEUROSCI.6776-10.2011.
34. Aleksander, I.; Stonham, T.J. Guide to pattern recognition using random-access memories. *Iee Journal on Computers and Digital Techniques* **1979**, *2*, 29–40. https://doi.org/10.1049/IJ-CDT.1979.0009.
35. Kahneman, D.; Frederic, S., 2002. https://doi.org/10.1017/CBO9780511808098.004.
36. Nechesov, A.; Ruponen, J. Empowering Government Efficiency Through Civic Intelligence: Merging Artificial Intelligence and Blockchain for Smart Citizen Proposals. *Technologies* **2024**, *12*, 271. https://doi.org/10.3390/technologies12120271.
37. Nechesov, A.; Dorokhov, I.; Ruponen, J. Virtual Cities: From Digital Twins to Autonomous AI Societies. *IEEE Access* **2025**, *13*, 13866–13903. https://doi.org/10.1109/ACCESS.2025.3531222.
38. Ruponen, J.; Dorokhov, I.; Barykin, S.E.; Sergeev, S.; Nechesov, A. Metaverse Architectures: Hypernetwork and Blockchain Synergy. In Proceedings of the First Conference of Mathematics of AI, 2025.
39. Dorokhov, I.; Ruponen, J.; Shutsky, R.; Nechesov, A. Time-Exact Multi-Blockchain Architectures for Trustworthy Multi-Agent Systems-. *MathAI* **2025**.

Short Biography of Authors

Andrey Nechesov— Corresponding Member of RIA, Head of Research Department at the AI Center of Novosibirsk State University. Main organizer of the international conference "Mathematics of Artificial Intelligence". Expert in the field of strong artificial intelligence, blockchain technologies, and cryptocurrencies. Author and creator of the investment-analytical strategy "One Way — One AI".

Janne Ruponen —Multidisciplinary Engineer and Researcher with a background in mechanical engineering, innovation, and technology management. His expertise spans the development of advanced manufacturing technologies, including significant contributions to additive manufacturing and 3D printing. He has also led impactful projects in urban planning and automation, such as creating unmanned ground vehicle (UGV) systems and digital twin implementations for heavy industries. His current research explores the integration of blockchain and artificial intelligence, with a focus on applications for moderative multi-agent systems.