Organic Learning Architectures: OLA, OLM, and OLC as a Continuous-Learning Alternative to Gradient-Ba

Abstract

Modern machine learning systems achieve impressive performance but rely on offline training, static weigh

This work presents an alternative architecture, built over several iterations of empirical experimentation, th

- OLA – the Organic Learning Architecture, an agent framework built around evolving genomes, intrinsic si
- OLM – the Organic Learning Model, a perceptual and predictive layer (e.g., VAEs and temporal models)
- OLC – the Organic Learning Cortex, a spiking, synaptic, continuously plastic substrate that implements m

The paper describes how these components emerged, how early versions were stressed and broken in pr

1. Introduction

Most contemporary AI systems are defined by three properties:

1. Static training regime: Models are trained offline on large datasets and then frozen.
2. Centralized memory in weights: All long-term knowledge is stored in dense parameters updated via grad
3. Lack of continuous adaptation: Once deployed, models do not learn in real time without retraining.

These properties make engineering straightforward but fundamentally conflict with the behavior of biologic

- learn continuously from sparse, noisy, and non-stationary streams,
- maintain long-lived memories without catastrophic erasure, and
- adapt without ever “restarting training.”

This work evolves an alternative: an Organic Learning Architecture (OLA) that treats an agent as a living c

- OLM, a perceptual module that compresses raw sensory input into usable latent codes;
- OLC, a synaptic, spiking cortex that provides continuous memory and temporal learning without gradient

The architecture did not arise from theory first. It emerged from iterative, often frustrating experiments on:

- genome-based logic systems solving math and game tasks,
- real-time agents playing environments like Snake,
- chat-shaping models that wrapped a frozen language model with an adaptive controller,
- continuous next-frame prediction systems (OLM) that performed well at one-step horizons but collapsed a

Through these tests, the core issue repeatedly reappeared: weight-based memory is unstable under contin

2. Background and Motivation

2.1 Continuous learning and representational instability

In a traditional gradient-based model, all knowledge is stored in weight matrices. Continuous online training

- the same weights are updated on every new sample,

- every new update can overwrite old information,
- distributional shifts translate directly into representational drift.

This is catastrophic forgetting in practice. Attempts to mitigate it (replay buffers, regularization, dual-networ

During the development of OLA and OLM, several recurring pathologies appeared:

- Horizon collapse in prediction: OLM models could predict the next frame or step, but performance degrad
- Attractor lock-in: Agents would fall into strong behavioral attractors; once a pattern was reinforced, the sy
- Non-local interference: Updates required for one task or environment configuration damaged behavior or

These behaviors suggested that weight-based representation, updated continuously, was fundamentally m

2.2 Biological learning as a design reference

Biological nervous systems do not store all knowledge in a single trainable matrix. Instead, they:

- represent information in spiking patterns,
- adjust synapses locally based on spike timing and neuromodulators,
- distribute memory across circuits and assemblies,
- separate timescales: fast synaptic changes, slower consolidation, ultra-slow structural/evolutionary chang

Existing research on spiking neural networks (SNNs) and Spike-Timing-Dependent Plasticity (STDP) show

However, such systems were historically limited by:

- lack of structured sensory input (raw pixels or toy symbolic sequences),
- small scale and toy tasks,
- limited integration with perception and action,
- poor tooling and weak hardware support in earlier decades.

The OLA/OLM/OLC stack aims to bridge this gap: using modern latent encoders (OLM) to provide structur

3. The OLA Stack: OLA, OLM, and OLC

3.1 OLA – Organic Learning Architecture (Agent Layer)

OLA is the agent-level framework responsible for:

- managing the organism’s state variables (energy, sleep, hunger, etc.),
- evolving and selecting genomes (discrete logic-like programs) that define action policies,
- integrating intrinsic signals such as trust, novelty, boredom, and survival metrics.

Key properties:

- Genomes as logic programs: Instead of a monolithic policy network, OLA uses genome-like structures en
- Trust-based selection: Genomes are evaluated via a “trust” signal: patterns that consistently lead to survi
- Exploration through mutation: Genomes mutate and recombine over time, allowing new strategies to be e

OLA was initially tested in:

- math tasks (e.g., simple equations where genomes mapped inputs to outputs),
- the Snake game (where genomes controlled movement strategies),
- chat shaping (where genomes influenced how a frozen language model responded).

These experiments showed that OLA could, in principle, evolve useful behaviors. However, without a stabl

3.2 OLM – Organic Learning Model (Perception and Prediction)

OLM is the perceptual and predictive layer, designed to:

- compress sensory streams (vision, text, audio) into low-dimensional latents,
- model local temporal structure (next-frame/next-step dynamics),
- provide a stable, structured input space for higher layers.

Typical OLM components include:

- Variational Autoencoders (VAEs) for vision: mapping raw frames to latent vectors, capturing spatial struct
- Temporal models (e.g., LSTMs or similar) for local sequence modeling: capturing short-term temporal co

Empirically, OLM worked well for:

- single-step prediction: given current latent, predict the next latent or frame,
- basic pattern extraction: primitive feature detectors emerged in the latent space.

However, in continuous learning settings, OLM inherited the same weight-based instabilities:

- multi-step predictions degraded rapidly over time,
- online updates caused latent representations to drift, breaking stability for downstream components,
- attempts to keep OLM plastic while maintaining performance led to either overfitting or collapse.

This indicated that OLM should be treated more like a sensory encoder (eyes/ears) than a central memory

3.3 OLC – Organic Learning Cortex (Synaptic Memory and Sequence Learning)

OLC is the new cortical layer designed specifically to address the continuous learning problem.

Core design

OLC is a recurrent network of spiking neurons with plastic synapses:

- Neurons: simple integrate-and-fire units with threshold and refractory periods.
- Synapses: directed connections with a persistent strength and a temporary plastic component.
- Dynamics: each time step, neurons integrate synaptic input, noise, and external stimulation; when a neur

Learning is implemented via a local STDP-like rule:

- If a pre-synaptic neuron fires shortly before a post-synaptic neuron, the synapse is potentiated.
- If the pre-synaptic neuron fires after the post-synaptic neuron, the synapse is depressed.
- A decay term gradually relaxes temporary plastic changes, while longer-term consolidation accumulates

This yields:

- local learning: updates depend only on spike timing at each synapse,
- multi-timescale memory: fast plasticity vs slower consolidation,
- distributed memory: sequences and associations are stored across many synapses and paths rather than

Tiered development

To progressively validate OLC, the system was developed in tiers:

- Tier 1 – Primitive synaptic organism:
  - 8–12 neurons in a small recurrent graph.
  - Random connectivity and noise.
  - Manual stimulation of selected input neurons via keyboard.
  - Visualization of neurons as nodes and synapses as color-coded edges in Pygame.
  - Goal: demonstrate stable spiking dynamics and visible synaptic differentiation.

- Tier 2 – Sequence learning and recall:
  - An automated test harness stimulated pairs or triplets of neurons in fixed temporal sequences during a t
  - In a recall phase, only the first neuron was stimulated (e.g., neuron 0), and the system observed whethe

From Weights to Synapses: Lessons from Failures

The transition from weight-based models (OLM with LSTMs, OLA controllers) to OLC was not planned fron

Early OLA/OLM experiments

Several classes of experiments were run:

- Math tasks: genomes in OLA tried to solve simple arithmetic equations, with a reward/trust signal for corr
- Game environments: OLA-controlled agents in environments like Snake, with trust and reward tied to sur
- Chat shaping: OLA operated as a latent controller around a frozen language model, influencing the style

Common observations:

- The systems could learn something—partial strategies, local behaviors, short stretches of coherent respo
- However, memory was brittle: as learning continued, new adaptations often destroyed older behaviors.
- Multi-step predictions and long-horizon planning repeatedly failed to stabilize.

The conclusion was that treating the LSTM/weight-based components as continuously plastic memory was

Realizing the weight substrate as the core bottleneck

A key shift in perspective occurred when:

- continuous learning was recognized as not just a “training problem,” but a representational stability proble
- the issue was reframed: “the architecture is new, but the memory substrate (weights) is old,”
- the model’s failures were traced back to using weights as the only form of memory.

Integrating OLM, OLC, and OLA

In the current design, the three components are arranged as follows:

1. OLM (Organic Learning Model): Perception
2. OLC (Organic Learning Cortex): Memory and Temporal Structure
3. OLA (Organic Learning Architecture): Action and Evolution

Experimental Status and Future Directions

So far, the following milestones have been reached:

- Stable synaptic organism (Tier 1)
- Sequence learning and recall (Tier 2, one-step)
- Test harness and analysis

Planned steps include multi-step sequence learning, sensory integration, action coupling, and eventually la

Conclusion

The OLA/OLM/OLC stack aims to step outside the limitations of gradient-based, static models by introduci