EMERGE

WP4

D4.2 First version of the ADS

Version: 1.0

Date: 31/03/2025

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Document control

Project title	Emergent awareness from minimal collectives
Project acronym	EMERGE
Call identifier	HORIZON-EIC-2021-PATHFINDERCHALLENGES-01-01
Grant agreement	101070918
Starting date	01/10/2022
Duration	48 months
Project URL	http://eic-emerge.eu
Work package	WP4
Deliverable	D4.2 First version of the ADS
Contractual delivery date	M30
Actual delivery date	M30
Nature[1]	Other
Dissemination level[2]	PU
Lead beneficiary	UNIPI
Editor(s)	Claudio Gallicchio
Contributor(s)	Andrea Ceni, Andrea Cossu, Claudio Gallicchio, Lucia Passaro, Valerio De Caro, Luigi Quarantiello, Lorenzo Leuzzi, Michael Biggeri, Gioele Zerini, Francesca Poli
Reviewer(s)	Cosimo Della Santina
Document description	This deliverable presents the final version of the Archetype Computing System (ACS) and the first version of the Archetype Adapting System (ADS), developed within WP4 of the EMERGE project. It consolidates the theoretical and experimental progress on archetype-based neural models, including the extension of Random Oscillator Networks (RON), their antisymmetric and physically implementable variants, and hybrid spiking-mechanical systems. Furthermore, it introduces adaptive learning strategies based on feedback alignment, equilibrium propagation, and lifelong evolutionary mechanisms. The deliverable also includes results on modular RNN assemblies and emergent awareness, along with an update of the ACDS software library supporting the implementation and evaluation of the ACS and ADS components.

[1] R: Document, report (excluding the periodic and final reports); DEM: Demonstrator, pilot, prototype, plan designs; DEC: Websites, patents filing, press & media actions, videos, etc.; DATA: Data sets, microdata, etc.; DMP: Data management plan; ETHICS: Deliverables related to ethics issues.; SECURITY: Deliverables related to security issues; OTHER: Software, technical diagram, algorithms, models, etc.

[2] PU – Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page); SEN – Sensitive, limited under the conditions of the Grant Agreement; Classified R-UE/EU-R – EU RESTRICTED under the Commission Decision No2015/444; Classified C-UE/EU-C – EU CONFIDENTIAL under the Commission Decision No2015/444; Classified S-UE/EU-S – EU SECRET under the Commission Decision

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Version control

Version[3]	Editor(s), Contributor(s), Reviewer(s)	Date	Description
0.1	Claudio Gallicchio	07/02/2025	Document created. TOC proposed.
0.3	Andrea Ceni, Andrea Cossu, Claudio Gallicchio, Lucia Passaro, Valerio De Caro, Luigi Quarantiello, Lorenzo Leuzzi, Michael Biggeri, Francesca Poli, Gioele Zerini	21/02/2025	First draft of the document.
0.5	Andrea Ceni, Andrea Cossu, Claudio Gallicchio, Lucia Passaro, Valerio De Caro, Luigi Quarantiello, Lorenzo Leuzzi, Michael Biggeri, Francesca Poli, Gioele Zerini	28/02/2025	Second draft of the document.
0.7	Andrea Ceni, Andrea Cossu, Claudio Gallicchio, Lucia Passaro, Valerio De Caro, Luigi Quarantiello, Lorenzo Leuzzi, Michael Biggeri, Francesca Poli, Gioele Zerini	07/03/2025	Third draft of the document
0.8	Claudio Gallicchio	14/03/2025	Document finished. Added Document Description, Abstract, Introduction and Conclusions sections.
0.85	Cosimo Della Santina	21/03/2025	Document reviewed
0.9	Claudio Gallicchio	28/03/2025	Document revised by the editor.
1.0	Davide Bacciu	31/03/2025	Document released.

No2015/444

[3] 0.1 – TOC proposed by editor; 0.2 – TOC approved by reviewer; 0.4 – Intermediate document proposed by editor; 0.5 – Intermediate document approved by reviewer; 0.8 – Document finished by editor; 0.85 – Document reviewed by reviewer; 0.9 – Document revised by editor; 0.98 – Document approved by reviewer; 1.0 – Document released by Project Coordinator.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Abstract

Awareness in biological agents has converging definitions when considering local states describing content-related consciousness from an agent-specific perspective. However, it becomes highly debated when it comes to global states. The issue magnifies when considering collectives of artificial agents. Several frameworks exist, all unsatisfactory in the limitations posed to agents' heterogeneity and disappearance of the local self into an integrated state.

Ultimately, existing frameworks are ineffective in explaining, facilitating, and supporting cooperative behaviours in artificial agents. The lack of a compelling theory of global awareness in AI is currently a significant barrier to the effective deployment of artificial agents in the real world.

EMERGE tackles this grand challenge by introducing the novel concept of collaborative awareness for collectives of minimal artificial beings. We will investigate how simple agents can develop a representation of their mutual existence, environment, and cooperative behaviour towards the realisation of tasks and goals.

EMERGE builds on a scenario of artificial beings with no shared language and constrained individual capabilities, which nevertheless leads to high-complexity behaviours at the collective level. Collaborative awareness becomes an emergent process supporting complex, distributed, and loosely coupled systems capable of high degrees of collaboration, self-regulation, and interoperability without predefined protocols.

EMERGE delivers a philosophical, mathematical, and technological framework that enables us to know how and where to allocate awareness to optimally achieve a goal through the collective. We will demonstrate EMERGE concepts on robotic use cases, with hints of the broader applicability of the framework to the Internet of Things, pervasive computing, and nanotechnologies. We will also investigate the ethical implications of collaborative awareness, focusing on moral responsibility, vulnerabilities, and trust.

Consortium

The EMERGE consortium members are listed below.

Organization	Short name	Country
Università di Pisa	UNIPI	IT
TU Delft	TUD	NL
University of Bristol	UOB	UK
Ludwig Maximilian University of Munich	LMU	DE
Da Vinci Labs	DVL	FR

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Disclaimer

Acknowledgement

This document is a deliverable of EMERGE project. This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement Nº 101070918.

Table of Contents

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

List of Figures

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

List of Tables

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

List of Abbreviations

ACS, ADS, ACDS	Archetype Computing System, Archetype Adaptive System, Archetype Computing and Adaptive System
aPI-RON	antisymmetric physically-implementable Random Oscillators Network
aRON	antisymmetric Random Oscillators Network
BPTT	Backpropagation through time
CA	Consortium Agreement
DFA	Direct Feedback Alignment
EP	Equilibrium Propagation
ESN	Echo State Network
euESN	Euler State Network
EWC	Elastic Weight Consolidation
GA	Grant Agreement
GRU	Gated Recurrent Unit
IPR	Intellectual Property Rights
LIF	Leaky-Integrate and Fire
MC	Memory Capacity
NEAT	Neuroevolution of Augmenting Topologies
PI-RON	physically-implementable Random Oscillators Network
RIM	Recurrent Independent Mechanism
RNN	Recurrent Neural Network
RON	Random Oscillators Network
S-RON	Spiking Random Oscillators Network
WP	Work Package

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Executive Summary

This document is a deliverable of the EMERGE project, funded under grant agreement number 101070918. This deliverable, D4.2, is part of work package Learning and Evolutionary Awareness – Adaptive non-linear dynamical systems for awareness (WP4), which focuses on the development of computational mechanisms enabling adaptive and learning behavior in archetype-based artificial systems.

D4.2 presents the final version of the Archetype Computing System (ACS) and introduces the first implementation of the Archetype Adapting System (ADS), completing the conceptual and technical foundations of the Archetype Computing and aDaptive System (ACDS) framework. It consolidates the theory and implementation of archetype networks through extensions of the Random Oscillators Network (RON), including antisymmetric and physically implementable variants, and their integration with spiking neural components. On the adaptation side, the deliverable introduces biologically inspired learning strategies such as Direct Feedback Alignment (DFA) and Equilibrium Propagation (EP), along with a framework for lifelong evolutionary adaptation in swarm systems.

In addition to theoretical and empirical contributions, the deliverable reports on the continued development of the ACDS software library, supporting experimentation and reproducibility. The associated software framework is publicly available at: `https://github.com/EU-EMERGE/archetype-computing-adaptive-system`.

1. Introduction

The EMERGE project aims to establish a novel computational framework inspired by the emergent properties of simple, interconnected systems, exploring how awareness emerges from the development and interaction of a collective of intelligent agents. As part of this broader initiative, Deliverable D4.2 builds directly on the foundational work laid out in Deliverable D3.2 (“Archetype Units and Connectors”) and Deliverable D4.1 (“First Version of the Archetype Computing System”), expanding these concepts into a robust and adaptive computing framework.

In Deliverable D3.2, the notion of archetype units and connectors was formalized, providing a set of fundamental building blocks for constructing networks of dynamical systems. Archetype units, such as oscillators, spiking neurons, and other interpretable dynamical models, serve as the basic computational elements. Connectors establish interactions between these units, enabling the formation of structured networks that we expect to be capable of complex information processing. This work established the groundwork for the EMERGE Archetype Computing and aDaptive System (ACDS), emphasizing the theoretical and practical aspects of assembling these units into cohesive networks. Figure 1 visually represents these concepts.

Building upon this, Deliverable D4.1 introduced the first iteration of the Archetype Computing System (ACS). This framework demonstrated how archetype networks could be employed to facilitate intelligent computation through the dynamic interaction of minimal, adaptable units. It explored various learning paradigms, including reservoir computing, physics-inspired neural networks, and algorithms beyond traditional backpropagation, laying the theoretical and experimental foundation for adaptive systems. The Random Oscillators Network (RON) model, a key innovation of D4.1, exemplified the potential of archetype networks in creating efficiently trainable recurrent neural architectures. Notably, the ACS framework proved to be more effective than standard methods in enabling efficient and adaptive learning, as evidenced for instance by the superior performance of the RON model compared to traditional recurrent

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

neural networks. A key visual representation of the WP4 work is provided in Figure 1, which

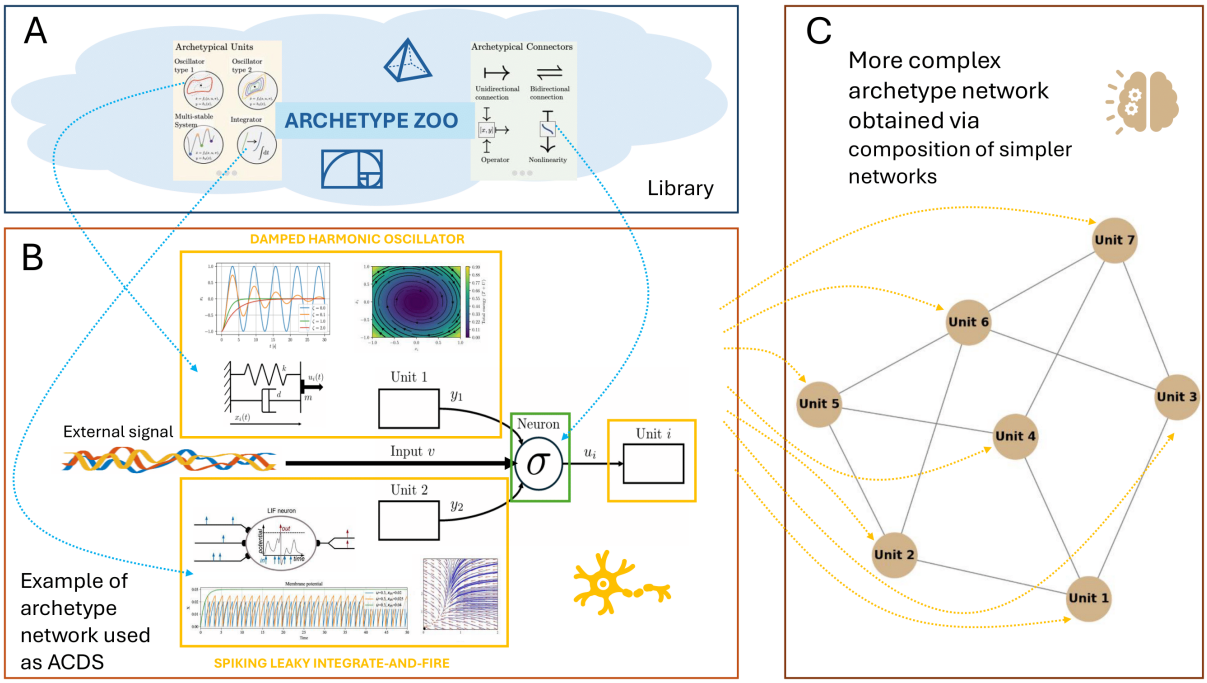


Figure 1: The core idea of the ACDS framework of EMERGE. The figure illustrates the construction of an archetype network, starting from the abstract conceptual space called the "Archetype Zoo" (represented by the blue cloud) in box A. This space contains fundamental archetypical units and connectors, akin to Platonic forms of dynamical systems. Archetypical units include oscillators, multi-stable systems, and integrators, while the connectors represent possible interactions such as unidirectional and bidirectional links, operators, and nonlinearities. From this abstract space, specific dynamical systems are instantiated in box B, such as the damped harmonic oscillator and the spiking leaky-integrate-and-fire (LIF) unit, both depicted in the yellow boxes. These systems capture core behaviors observed in physical and biological systems, like oscillatory dynamics and spiking activity. An external signal, potentially representing real-world sensory data, feeds into the system and influences the behavior of individual downstream units (akin to the neuron at the end of the axon of a biological neuron). The systems are connected through instantiations of specific connectors, e.g. a neuron-like connector in the figure (highlighted in green), which integrates the outputs of different units into new inputs, shaping the overall system dynamics. This leads to the formation of an archetype network, composed of multiple interconnected units in box C (e.g., Unit 1 to Unit 7). The network topology, i.e how units are connected, plays a critical role in determining the collective behavior and performance of the system. Different network topologies yield distinct dynamical properties. For instance, skew-symmetric topologies promote marginal stability, enabling long-term information propagation, while hierarchical topologies foster the development of different timescales across layers, supporting structured and efficient information processing. By tailoring the composition and arrangement of these archetypical elements, the archetype network can be optimized for specific computational tasks, blending theoretical elegance with practical functionality.

serves as a guideline for understanding the transformation of archetype units and connectors into sophisticated networks designed for intelligent computation. The figure encapsulates

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

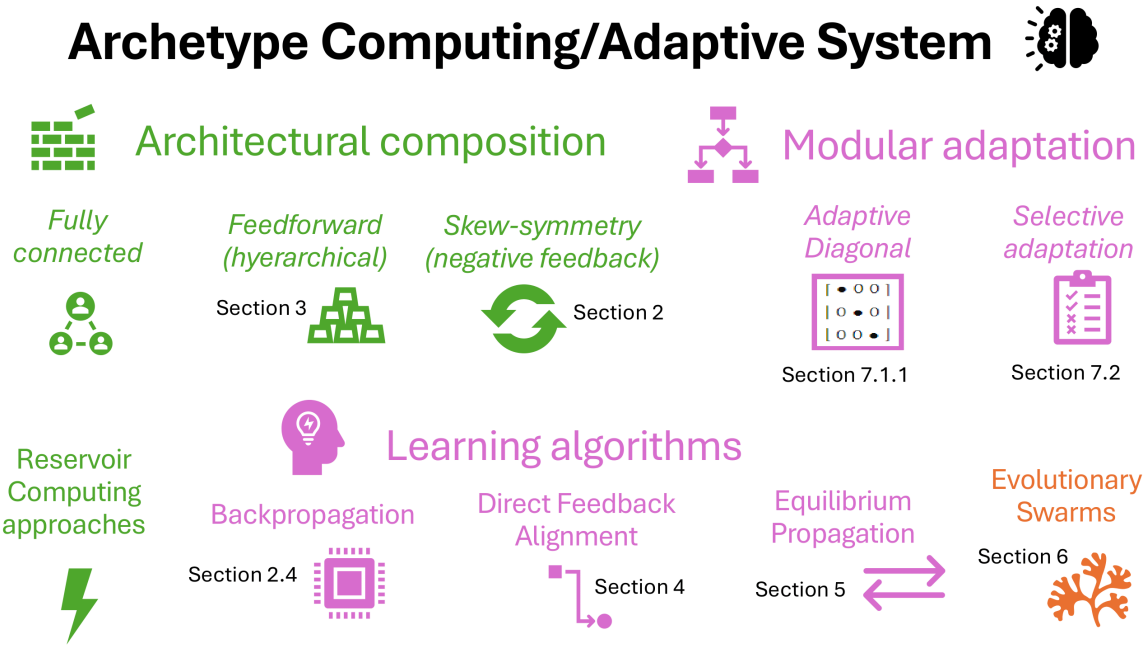


Figure 2: A visual summary of the progress and key developments of Deliverable D4.2. Advancements in Task 4.1 are highlighted in green, Tasks 4.2 and 4.3 in purple, and Task 4.4 in orange.

the essential elements of the Archetype Zoo, where fundamental dynamical systems like oscillators, multi-stable systems, and integrators are categorized, and the types of connections that link them, including unidirectional, bidirectional, and nonlinear operators. We build neural network architectures capable of solving downstream tasks by integrating external signals with, for example, archetypal units such as the Damped Harmonic Oscillator and the Spiking Leaky Integrate-and-Fire (LIF) neuron. This visual synthesis also highlights the role of modular networks, with interconnected units forming an adaptive system capable of complex behaviors. By mapping the flow of signals and the dynamic transformations occurring within the network, Figure 1 provides a foundational overview of the architecture and function of the EMERGE computational framework.

Deliverable D4.2 extends these efforts by presenting the first version of the Archetype aDaptive System (ADS). We refer to ACS and ADS together as ACDS (Archetype Computing and aDaptive System). This document details the transition from static computational structures to adaptive, evolving networks capable of lifelong learning and real-time adaptation. Central to this advancement is the transformation of archetype units and connectors into scalable networks, optimized for task-oriented behavior.

The subsequent sections of Deliverable D4.2 are organized to reflect the progression from foundational models to sophisticated adaptive frameworks, following the visual outline provided in Figure 2. This figure serves as a roadmap for the document, structuring its content into distinct yet interconnected themes. It categorizes the key aspects of the ACDS into three main areas: Architectural Composition, Modular Adaptation, and Learning Algorithms. Under Architectural Composition, it depicts various network configurations such as fully connected architectures, hierarchical feedforward systems, and the use of skew-symmetry to enforce negative feedback for system stability. Modular Adaptation is illustrated through methods like adaptive diagonal matrices and selective adaptation strategies, emphasizing the system's

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

capacity to adjust its parameters in response to changing environments. Finally, the figure outlines the core learning algorithms explored in the document, including Reservoir Computing approaches, Backpropagation, Direct Feedback Alignment, and Equilibrium Propagation, each chosen for their potential to support efficient and biologically inspired learning in archetype networks.

Final version of the ACS. The ACS has been completed by integrating several archetype units, connectors, and networks.
Specifically, Section 2 introduces Anti-symmetric Random Oscillator Networks (aRON), an extension of the RON architecture designed to improve stability and facilitate physical implementation. This section explores the theoretical underpinnings of aRON, providing conditions for stability and showcasing experimental results. **aRON is an archetype network composed of harmonic oscillators archetype units** (Section 2.2 of D3.2) interacting with each other through a **Neuron-like Connector** (Section 3.6 of D3.2) with a global **skew-symmetric structure in the coupling of the units** (see also Section 4.4.1 of D3.2). We also considered and implemented the **Physically-implementable aRON** (Section 2.4), leveraging antisymmetric connections for the physically-implementable version of RON.
Section 3 integrates Random Oscillator Networks with spiking archetypes, forming the S-RON archetype network. This hybrid approach leverages the strengths of both continuous and event-driven dynamics. The section also outlines future directions for refining these hybrid models. **S-RON is an archetype network made of a layer of spiking Leaky Integrate-and-Fire (LIF) units**, as detailed in Section 2.10 of D3.2, which project feedforwardly into a subsequent layer of **harmonic oscillators** (Section 2.2 of D3.2). The connections from spiking units to oscillator units are modeled by the **Threshold Connector**, as described in Section 3.4 of D3.2, which includes a reset mechanism, and connections from oscillators to spiking units are modeled by a **Series Connector** as described in Section 3.1 of D3.2.

First version of the ADS. The ADS provides the necessary component to adapt archetype units and networks.
Section 2.4 shows how the antisymmetric physically-implementable RON can be effectively adapted to a wide range of tasks with excellent results.
Sections 4 and 5 deal with **learning paradigms alternative to backpropagation** that are included in the ADS. In particular, Section 4 focuses on Feedback Alignment methods. It discusses Direct Feedback Alignment (DFA) for both feedforward and recurrent networks, presenting experimental results and a critical analysis of its efficacy. Our work on Direct Feedback Alignment for recurrent networks is currently under review at the 5th International Workshop on Computational Aspects of Deep Learning within the 2025 ISC High Performance (Hamburg). Section 5 introduces Equilibrium Propagation, an alternative learning strategy grounded in physical principles. This section evaluates its application within archetype networks, highlighting its potential for biologically plausible and energy-efficient learning.
Section 6 describes how lifelong learning can be implemented through evolutionary adaptation. We focus on the case of robotic swarms and we introduce the **lifelong evolutionary swarms framework, presenting an environment where distributed agents continuously adapt through evolutionary algorithms**. This section details the experimental setup, examines quick adaptability and memory retention, and addresses challenges such as catastrophic forgetting and stress resilience. Our work on lifelong evolutionary swarms has been recently accepted at the 2025 GECCO conference (Málaga).

The ACS and the ADS are implemented in the ACDS library, described in Section 9. The ACDS

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

library provides essential tools for implementing and experimenting with archetype networks.

In addition to the key advances in the ACS and ADS described above, we also describe some preliminary results:

- Section 7 presents preliminary results on **training modular ensembles of Recurrent Neural Networks (RNNs)**. It introduces the **framework for collective negative-feedback coupling**, with advancements in trainable modular architectures and the **integration of attention mechanisms**.
- Section 8 advances the study of emergent awareness through modular RNN composition, investigating both temporal and spatial awareness in distributed networks.

We conclude the document in Section 10 by synthesizing the insights gained and outlining future research directions. Deliverable D4.2 is a key step in the EMERGE project, bridging theoretical constructs with practical implementations in realizing adaptive, intelligent collective systems.

2. Anti-symmetric Random Oscillator Networks

Random Oscillators Network (RON), as introduced in Section 3 of D4.1, is an archetype network made of harmonic oscillators (Section 2.2 of D3.2) coupled by a neuron-like connector (Section 3.6 of D3.2). The novelty of aRON lies in the topological structure of the coupling of the archetype units, at the network level. We employ a skew-symmetric network structure. The rationale of this network is that it is biased towards the identity and it is more stable because of results in the literature of skew-symmetric couplings. Moreover, it has physical meaning in terms of control systems because skew-symmetric coupling of two units equals connecting the two units in negative-feedback. We provide theoretical evidence of stability of the model and provide extensive experimental data showing the effectiveness of aRON in the classification of time series.

2.1. The aRON model

We design an antisymmetric coupling for the units of the RON model, resulting in the *Antisymmetric RON* (aRON). The idea behind an antisymmetric coupling is to endow the RON model with increased ability to stably propagate information over long time spans. The antisymmetric coupling is commonly present in many physical systems, where the modules to be coupled are in negative feedback. Intuitively, in the presence of antisymmetric coupling, the activations of a given module are inversely proportional to the activations of the module(s) it is connected to (Figure 3).

Specifically, we introduce the discrete-time state-update equation of antisymmetric-RON (aRON), as follows:

$$\mathbf{h}_{t+1} = \mathbf{h}_t + \tau \mathbf{z}_{t+1}, \tag{1}$$

$$\mathbf{z}_{t+1} = \mathbf{z}_t + \tau(\sigma(\mathbf{A}\mathbf{h}_t + \mathbf{V}\mathbf{u}_{t+1} + \mathbf{b}) - \gamma \odot \mathbf{h}_t - \varepsilon \odot \mathbf{z}_t), \tag{2}$$

where $\mathbf{A} = (\mathbf{W} - \mathbf{W}^T) - \delta\mathbf{I}$, is parametrized as in the EuSN model (see D4.1), and the remaining components as in the RON model.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

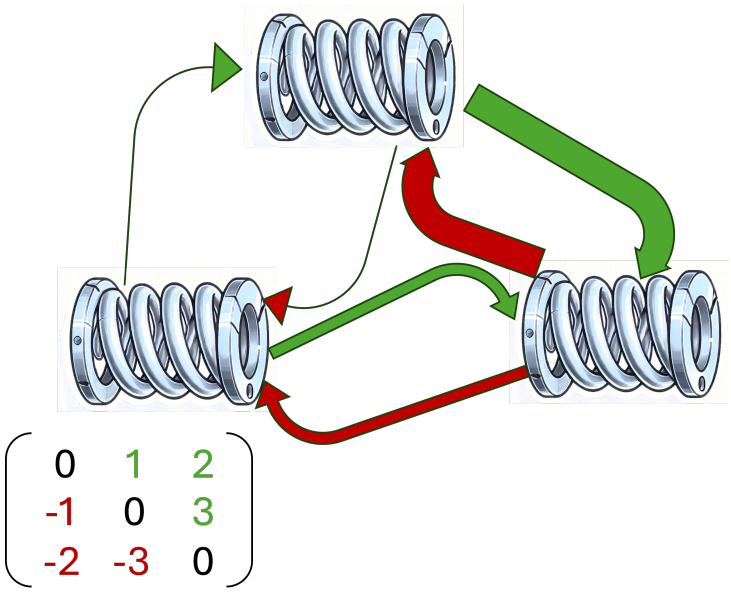


Figure 3: Physical systems in negative feedback coupling. The matrix regulating the connections between the mechanical springs is antisymmetric ($\mathbf{A} = -\mathbf{A}^T$). The width of the arrows represents the connection strength, the positive feedback is green, and the negative feedback is red. Each spring receives a signal from the other springs of the same magnitude as the signal it sends but of inverse sign.

2.2. Necessary condition for stability of aRON

Let us denote $\mathbf{H}_t = \begin{pmatrix} \mathbf{h}_t \\ \mathbf{z}_t \end{pmatrix}$. Then, the aRON model can be defined by the input-driven state-update equation $\mathbf{H}_{t+1} = G(\mathbf{H}_t, \mathbf{u}_{t+1})$, where $G : \mathbb{R}^{2H} \times \mathbb{R}^I \to \mathbb{R}^{2H}$ is defined by eqs. (1) - (19). In the remainder of this section, we assume that $\mathrm{diag}(\boldsymbol{\varepsilon}) = \varepsilon\mathbf{I}$ and $\mathrm{diag}(\boldsymbol{\gamma}) = \gamma\mathbf{I}$ are scalar matrices. The Jacobian of the G map computed on $(\mathbf{H}_t, \mathbf{u}_{t+1})$, denoted with $\mathbf{J}_k$, reads:

$$\mathbf{J}_t = \begin{bmatrix} \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{h}_t} & \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{z}_t} \\ \frac{\partial \mathbf{z}_{t+1}}{\partial \mathbf{h}_t} & \frac{\partial \mathbf{z}_{t+1}}{\partial \mathbf{z}_t} \end{bmatrix} = \tag{3}$$

$$= \begin{bmatrix} \mathbf{I} + \tau^2 \mathbf{M}_t & \tau\big(1 - \tau\varepsilon)\mathbf{I}\big) \\ \tau\mathbf{M}_t & \big(1 - \tau\varepsilon)\mathbf{I} \end{bmatrix}, \tag{4}$$

where

$$\mathbf{M}_t = \mathbf{D}_t(\mathbf{W} - \mathbf{W}^\top - \delta\mathbf{I}) - \gamma\mathbf{I}, \tag{5}$$

$$\mathbf{D}_t = \mathrm{diag}\big(\sigma'((\mathbf{W} - \mathbf{W}^\top - \delta\mathbf{I})\mathbf{h}_t + \mathbf{V}\mathbf{u}_{t+1} + \mathbf{b})\big). \tag{6}$$

A widely known necessary condition for stability of reservoir computing systems is to impose that the Jacobian evaluated at the origin, without input and without bias, has a spectral radius close to 1. This condition ensures that, in the absence of input, the autonomous dynamics of the system are asymptotically stable. Notice that, when using $\sigma = \tanh$ as nonlinearity,

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

evaluating at the origin, without input, and without bias, then the diagonal matrix of (6) is the identity matrix. Therefore, the Jacobian reads:

$$\mathbf{J}_0 = \begin{bmatrix} \mathbf{I} + \tau^2(\mathbf{W} - \mathbf{W}^\top - \delta\mathbf{I} - \gamma\mathbf{I}) & \tau\big(1 - \tau\varepsilon\big)\mathbf{I} \\ \tau(\mathbf{W} - \mathbf{W}^\top - \delta\mathbf{I} - \gamma\mathbf{I}) & \big(1 - \tau\varepsilon\big)\mathbf{I} \end{bmatrix}, \tag{7}$$

In the proposition below we estimate the location of the eigenvalues of $\mathbf{J}_0$, whose proof can be found in Appendix A.

Proposition 1. *If μ is an eigenvalue of $\mathbf{J}_0$, then μ is at distance at most $\tau \max\{|1-\tau\varepsilon|, \sqrt{\lambda_{\max}^2 + (\delta+\gamma)^2}\}$ from the set of values $\{1-\tau\varepsilon\} \cup \{1 - \tau^2(\delta + \gamma \pm i\lambda) : i\lambda$ is an eigenvalue of $\mathbf{W} - \mathbf{W}^\top\}$, where λ_{max} is the spectral radius of the skew-symmetric matrix $\mathbf{W} - \mathbf{W}^\top$.*

The set of values $\{1-\tau\varepsilon\} \cup \{1 - \tau^2(\delta + \gamma \pm i\lambda) : i\lambda$ is an eigenvalue of $\mathbf{W} - \mathbf{W}^\top\}$ effectively provide an estimation for the location of the eigenvalues of $\mathbf{J}_0$, especially accurate for small values of τ. Therefore, a necessary condition for aRON stability is that the set of these values is completely contained inside the unit circle. In the theorem below, we formally write a few inequalities that aRON's hyperparameters must necessarily satisfy for stability. The proof can be found in Appendix B

Theorem 1. *Assume an aRON model with $\delta, \tau, \varepsilon, \gamma > 0$. If the aRON of eq.s* (1)-(19) *is stable, then*

$$\varepsilon \leq \frac{2}{\tau}, \tag{8}$$

$$\delta + \gamma \leq \frac{2}{\tau^2}. \tag{9}$$

Moreover, each eigenvalue $i\lambda$ of the skew-symmetric matrix $\mathbf{W} - \mathbf{W}^\top$ must have modulus upper-bounded as follows:

$$|\lambda| \leq \tau\sqrt{(\delta+\gamma)\big[2 - \tau^2(\delta+\gamma)\big]}. \tag{10}$$

We can see from eq. (10) that, for a given setting of τ, γ, positive values of the diffusion term δ can facilitate the necessary condition of stability to hold. In this sense, larger values of δ can help the stability of the aRON, but at the cost of a higher dissipation rate of the information flow. On the other hand, values too large of δ, e.g. $\delta \geq \frac{2}{\tau^2} - \gamma$, can cause instabilities. Therefore, δ must be carefully tuned for the specific task at hand.

We conclude with the following proposition that gives us a criterion on the norm of $\mathbf{W}$ to enforce the necessary condition for the stability of aRON. The proof can be found in Appendix C.

Proposition 2. *A strategy to promote the stability of aRON model is to clip the norm of $\mathbf{W}$ when reaching the following threshold value:*

$$||\mathbf{W}|| \leq \frac{1}{2}. \tag{11}$$

We use the insights from Theorem 1 to guide the model selection of aRON in the experimental section.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Table 1: Average accuracy and standard deviation over 5 runs obtained from each model on the considered benchmarks.

Model	Leaky-ESN	aESN	RON	aRON	EuSN	GRU
Adiac	$0.6928_{0.0116}$	$0.6793_{0.0099}$	$0.7090_{0.0125}$	$\mathbf{0.7315_{0.0043}}$	$0.4221_{0.0154}$	$0.6322_{0.0235}$
sMNIST	$0.8798_{0.0022}$	$0.8803_{0.0040}$	$0.9568_{0.0020}$	$\mathbf{0.9627_{0.0006}}$	$0.6321_{0.0401}$	$0.9686_{0.0165}$
psMNIST	$0.8284_{0.0272}$	$0.9067_{0.0054}$	$0.8760_{0.0050}$	$\mathbf{0.9106_{0.0065}}$	$0.4380_{0.0244}$	$0.8790_{0.0085}$
Trace	$0.9640_{0.0206}$	$0.9740_{0.0080}$	$0.9920_{0.0075}$	$\mathbf{0.9940_{0.0049}}$	$0.8200_{0.0228}$	$0.9979_{0.0040}$
Mallat	$0.9011_{0.0097}$	$0.9179_{0.0162}$	$0.9142_{0.0274}$	$\mathbf{0.9179_{0.0230}}$	$0.3364_{0.0306}$	$0.2142_{0.0021}$
LIBRAS	$0.7911_{0.0083}$	$0.7489_{0.0147}$	$0.7900_{0.0151}$	$\mathbf{0.7956_{0.0166}}$	$0.7722_{0.0136}$	$0.8344_{0.0221}$

Table 2: Number of trainable parameters for each benchmark

Benchmarks	ADIAC	sMNIST	psMNIST	Trace	Mallat	LIBRAS
Parameters	$\approx 3.7k$	$\approx 10k$	$\approx 10k$	$\approx 0.2k$	$\approx 0.8k$	$\approx 2.25k$

2.3. Experiments

The models considered in this study were evaluated on six benchmarks with varying characteristics: ADIAC, Trace, Mallat and LIBRAS, which primarly involve tasks with short-term dependencies and sMNIST and psMNIST, which are designed to assess the models' ability to handle long-term dependencies. This selection of benchmarks was motivated by prior studies demonstrating the advantages of using antisymmetric models, particularly in task involving long-term dependencies. In addition, by including benchmarks with short-term dependencies, this study aims to provide a more comprehensive evaluation of the models' versatility across a broader range of temporal dynamics.

Across all benchmarks, the reservoir computing models we evaluated (ESN, aESN, RON, aRON, and EuSN) were configured to use the same number of hidden units, ensuring a fair comparison of their structural characteristics. Specifically, the hidden layer sizes were set to: 50 units for Trace, 100 units for ADIAC and Mallat, 150 units for LIBRAS and 1000 units for sMNIST and psMNIST. Furthermore, to mantain comparability with the previous architectures, the fully-trained GRU was configured with a number of hidden units resulting in a similar number of trainable parameters as the other models (Table 2). The corresponding hidden unit sizes for the GRU were as follows: 28 units for ADIAC, 14 units for Mallat, 23 units for LIBRAS, 6 units for Trace and 55 units for sMNIST and psMNIST.

To maximize the models' performance, hyperparameter values were selected through an extensive grid search. RON and its antisymmetric variant (aRON) share the same set of hyperparameters, with a key distinction: the classical RON uses the spectral radius (ρ) to scale the hidden-to-hidden weight matrix, while aRON incorporates a diffusive term to stabilize the computation. Similarly, for ESN and its antisymmetric counterpart (aESN), both models have comparable set of hyperparameters; however, unlike RON, the ESN does not include the spectral radius as a tuning parameter.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

After this phase of model selection, each model was tested on five different runs, using the same hyperparameter combination but varying the weight initialization. The results presented in Table 1 show the average accuracies obtained across this runs, with the standard deviation indicated as a subscript.

The results are displayed in Table 1. The best performance achieved for each benchmark is highlighted in bold to facilitate comparisons.

From the results, it is evident that the antisymmetric version of the RON (aRON) consistently outperform all other reservoir computing models across the benchmarks. Notably, aRON significantly narrows the performance gap with fully-trained models, such as GRU. These findings underscore the efficacy of introducing antisymmetric structures in reservoir computing, demonstrating their potential as a promising direction for future research.

Additionally, the advantages of aRON are particularly pronounced in benchmarks designed to test long-term dependencies, such as sMNIST and psMNIST. These results align with the initial hypothesis that antisymmetric models would excel in tasks requiring the maintenance of information over extended time steps. The improved performance in these benchmarks not only validates the hypothesis but also highlights the robustness of antisymmetric reservoirs in handling complex sequential data.

2.4. Physically implementable and Antisymmetric Random Oscillator Networks

In D4.1, Section 3.5, we introduced the Physically-implementable RON (PI-RON), designed with TUD. Here, we briefly recall the model state update equation:

$$\mathbf{h}_{t+1} = \mathbf{h}_t + \tau \mathbf{z}_{t+1} \tag{12}$$

$$\mathbf{z}_{t+1} = \mathbf{z}_t + \tau(\mathbf{W}^T \sigma(\mathbf{V}\mathbf{u}_{t+1}) - \sigma(\mathbf{W}\mathbf{h}_t + \mathbf{b}) - \mathbf{\Gamma}\mathbf{h}_t - \boldsymbol{E}\mathbf{z}_t). \tag{13}$$

Similar to RON, we experimented also with the antisymmetric coupling of the PI-RON, where the hidden-to-hidden matrix $\mathbf{W}$ is parameterized as $(\mathbf{W} - \mathbf{W}^T) - \lambda\mathbf{I}$.

We train the matrices $\mathbf{V}, \mathbf{\Gamma}, \boldsymbol{E}$ and the bias $\mathbf{b}$ via backpropagation. Table 4 reports the number of trainable parameters for each dataset we used. Table 3 reports the accuracy obtained by PI-RON on each dataset.

The results show that PI-RON does not degrade the performance with respect to RON, despite its stronger constraints required to make it physically implementable. Moreover, the antisymmetric coupling of the units often results in a performance improvement. This strengthens the results previously showed for the antisymmetric RON model.

3. Integrating Random Oscillator Networks with spiking archetypes

In the RON model, a sigmoidal neuron-like transformation is used to excite the harmonic oscillator archetype units. We implement an archetype network with oscillator-based archetype units which, differently from RON, are excited by a layer of spiking Leaky Integrate-and-Fire (LIF) units, as detailed in Section 2.10 of D3.2. The connections from spiking units to oscillator units are modelled by the Threshold Connector, as described in Section 3.4 of D3.2, which includes a reset mechanism, and connections from oscillators to spiking units are modelled by a Series Connector as described in Section 3.1 of D3.2. We describe in detail the equation

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Model	Leaky-ESN	RON	aRON	PI-RON	aPI-RON
Adiac	$0.6928_{0.0116}$	$0.7090_{0.0125}$	$0.7315_{0.0043}$	$0.7278_{0.0063}$	$0.7309_{0.0082}$
sMNIST	$0.8798_{0.0022}$	$0.9568_{0.0020}$	$0.9627_{0.0006}$	$0.9504_{0.0032}$	$0.9566_{0.0037}$
Trace	$0.9640_{0.0206}$	$0.9920_{0.0075}$	$0.9940_{0.0049}$	$0.9800_{0.0151}$	$0.9700_{0.0127}$
Mallat	$0.9011_{0.0097}$	$0.9142_{0.0274}$	$0.9179_{0.0230}$	$0.9264_{0.0017}$	$0.9338_{0.0063}$
LIBRAS	$0.7911_{0.0083}$	$0.7900_{0.0151}$	$0.7956_{0.0166}$	$0.8156_{0.0138}$	$0.8256_{0.0103}$

Table 3: Accuracy of PI-RON on all datasets against RON and the ESN.

Dataset	ADIAC	sMNIST	Trace	Mallat	LIBRAS
#Parameters	$\approx 3.7k$	$\approx 10k$	$\approx 0.2k$	$\approx 0.8k$	$\approx 2.25k$

Table 4: Number of trainable parameters of PI-RON for each dataset.

ruling the dynamics of the model, and provide an empirical evaluation of the performance on time-series benchmarks. Then, a subsection discusses future directions for Mixed-RON and Spiking-Mechanical RON

3.1. The S-RON model

The second-order ODE of RON describes a pool of heterogeneous oscillators as follows:

$$\ddot{\mathbf{y}} = \mathbf{f}(t) - \boldsymbol{\gamma} \odot \mathbf{y} - \boldsymbol{\varepsilon} \odot \dot{\mathbf{y}}, \tag{14}$$

where $\odot$ denotes the point-wise multiplication of vectors, $\boldsymbol{\gamma}, \boldsymbol{\varepsilon} \in \mathbb{R}^N$ are the vectors collecting all γ and ε for each of the oscillators. Similarly, $\mathbf{y}, \dot{\mathbf{y}} \in \mathbb{R}^N$ collect all position and velocity for each of the oscillators. In the original RON model we have a sigmoidal layer that excites the harmonic oscillators:

$$\mathbf{f}(t) = \tanh(\mathbf{W}\mathbf{y} + \mathbf{V}\mathbf{u}(t) + \mathbf{b}), \tag{15}$$

and $\mathbf{u}(t)$ is the external input driving the network.

Here we consider replacing the sigmoidal layer of RON with a layer of spiking archetype units of the Leaky Integrate-and-Fire kind, where the dynamics of $\mathbf{f}(t)$ are ruled by:

$$\theta\dot{\mathbf{f}} = -\mathbf{f} + \mathbf{W}\mathbf{y} + \mathbf{V}\mathbf{u}(t) + \mathbf{b}, \tag{16}$$

where θ is the resistor-capacitor time constant, and with a hard-threshold reset mechanism as follows:

$$\text{if } \mathbf{f}_i \geq f_{\text{thresh}}, \quad \text{then } \mathbf{f}_i = f_{\text{reset}}, \tag{17}$$

where $\mathbf{f}_i$ denotes the i-th component of the spiking dynamical vector $\mathbf{f}(t)$, while f_{thresh} is the threshold value that activates a spike, and f_{reset} is the value $\mathbf{f}_i$ is reset after the i-th neuron fires.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

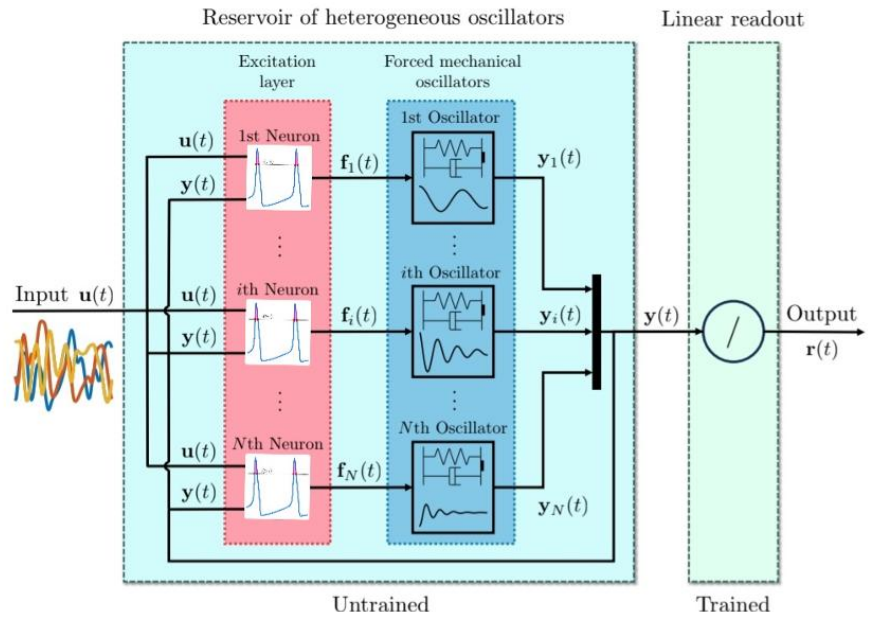


Figure 4: The Spiking Random Oscillators Network (S-RON) consists of N harmonic oscillators forced by coupled spiking LIF neurons in a feedforward fashion with hard-threshold reset mechanism. A linear output layer maps the states of the mechanical oscillators in the desired output. This layer is the only one that is adapted during learning.

The discrete-time realization of such a system reads:

$$\mathbf{h}_{t+1} = \mathbf{h}_t + \tau \mathbf{z}_{t+1}, \tag{18}$$

$$\mathbf{z}_{t+1} = \mathbf{z}_t + \tau(\mathbf{f}(t) - \gamma \odot \mathbf{h}_t - \varepsilon \odot \mathbf{z}_t), \tag{19}$$

We call the resulting archetype network Spiking-RON (S-RON). Therefore, S-RON is an archetype network comprehending both spiking and harmonic archetype units.

3.2. Preliminary Results and Future Directions for S-RON

Extensive grid search was conducted to determine the optimal parameters for the S-RON architecture on the sMNIST classification task. The best-performing parameters identified are reported in Table 5. Using the hyperparameters of Table 5, the S-RON achieved training and

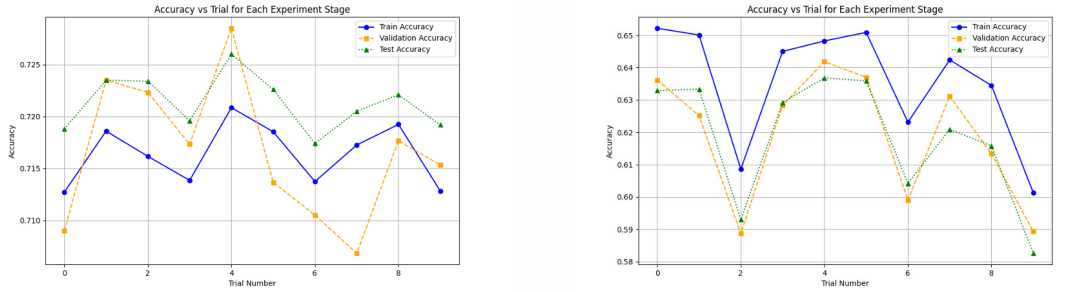
Model selection	τ	threshold	θ	reset
Best value	0.02	0.008	0.035	0.004

Table 5: Best S-RON hyperparameters found by grid search for the sMNIST classification task.

test accuracies of 0.65 and 0.64, respectively, on the sMNIST classification task. These results demonstrate the model's capability to capture complex dynamics through its spiking-based excitation layer.

When evaluated on the MNIST dataset, S-RON exhibited a slight drop in accuracy compared to the traditional RON architecture. Despite this minor reduction in accuracy, S-RON offers several advantages due to its spiking neuron dynamics. By employing a biologically inspired

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.



(a) RON accuracy (b) S-RON accuracy

Figure 5: Comparison of accuracy across trials for (a) RON and (b) S-RON models.

leaky integrate-and-fire mechanism, S-RON models neuronal behavior more realistically and enables energy-efficient computation. The use of spikes allows the model to better capture temporal dynamics and sparse activation patterns, making it well-suited for time-series tasks and neuromorphic computing applications. These unique features position S-RON as a promising alternative to the traditional RON despite the slight compromise in accuracy.

The results presented in Figure 5 provide a visual comparison between the performance of the RON and S-RON models. RON consistently achieves higher accuracy across training, validation, and test stages, maintaining performance around the 0.72 range. In contrast, S-RON displays slightly lower accuracy, peaking at 0.65 for training and around 0.64 for test accuracy. Despite this drop, S-RON's advantage lies in its biologically inspired mechanism, using spiking neurons that enable sparse, efficient computation more easily deployable in neuromorphic scenarios. The fluctuations in S-RON's performance are a reflection of the spiking behavior's sensitivity to parameter tuning and the timing of spikes, which introduces variability but also opens opportunities for more nuanced modeling of sequential data. Additionally, S-RON's lower energy usage and closer alignment with real neuronal activity position it as a more efficient alternative for neuromorphic hardware applications despite the slight trade-off in accuracy.

Future directions on hybrid oscillator-spiking models Ongoing and future work on S-RON focuses on fine-tuning and testing the model on additional benchmarks, such as the Mackey-Glass forecasting task. Efforts are also being directed toward smoothing the dynamic response of the S-RON to better align with the original RON's behavior. This optimization aims to leverage the architecture's potential for more efficient and robust time-series modeling. The next phase of research will also explore the integration of computational layers with a variable mixture of spiking LIF neurons and damped harmonic oscillators to further enhance the computational power and versatility of the model.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

4. Learning via Feedback Alignment

Backpropagation (Rumelhart, G. E. Hinton, and Williams 1986) is the long-standing algorithm for credit assignment in artificial neural networks. Its efficient implementation in digital computers has supported the surge of machine and deep learning techniques as one of the key advancements in the field of artificial intelligence (LeCun, Bengio, and G. Hinton 2015). However, with a few exceptions (Wright et al. 2022), the adoption of backpropagation-based learning systems is still mainly limited to digital computers and simulations. It is well known that backpropagation cannot be easily implemented and deployed in physical systems (Momeni et al. 2023; Lillicrap, Santoro, et al. 2020). Physical deployment of backpropagation is even more challenging in RNNs (Elman 1990), where credit assignment must be performed across time through BackPropagation Through Time (BPTT) (Werbos 1990). This issues limit the extent to which archetype networks can leverage backpropagation to adapt their dynamics. Over time, several backpropagation-free algorithms have been proposed (see Section 4.1 for a non-exhaustive overview), some of them with the explicit objective of being compatible with the implementation in physical systems or on unconventional hardware (e.g., neuromorphic, optical).

We focus on Direct Feedback Alignment (DFA) (Nøkland 2016), a backpropagation-free algorithm for credit assignment that removes the weight transport issue and also allows parallel computation of the weight update. DFA has already been implemented in nonconventional hardware, especially photonic (Filipovich et al. 2022). The photonic co-processor introduced in Launay et al. 2020 scales DFA to trillion-parameter random projections.

We briefly review DFA for feedforward networks in Section 4.2. We propose an extension of DFA tailored to recurrent neural networks. Our approach is able to compute the update of the recurrent parameters in parallel over all the time steps of the input sequence, thus removing one of the major drawbacks of BPTT. In fact, BPTT sends the error signal computed at the end of the input sequence *back in time* to compute the network parameters update. Instead, the update computed by our version of DFA is local at each time step, as it does not rely on the update computed for other time steps. Due to the weight sharing present in RNNs, the local update is eventually aggregated at the end of the input sequence to compute the final update. The aggregation operation includes information from all the time steps, thus enabling learning of temporal dependencies.

We develop DFA for both a “Vanilla” RNN and a Gated Recurrent Unit (GRU) network (Cho et al. 2014; Chung et al. 2014). We benchmark both architectures against BPTT on four time-series classification datasets and we find that DFA can achieve non-trivial performances in all of the tested datasets but cannot always attain a performance comparable to BPTT. In general, DFA shows strength in datasets with more than 2 classes and in datasets with a limited number of training samples, although BPTT still surpasses its performance. We show that the GRU architecture trained with DFA is able to learn longer temporal correlations than a “Vanilla” RNN.

4.1. Related works

Lillicrap, Cownden, et al. 2016 proposed the Feedback Alignment algorithm (FA) as a biologically plausible gradient-free learning rule for deep learning. The key idea of FA is to project the errors from the last layer of a deep feedforward architecture to the first layer via random projections between consecutive layers. This simple algorithm has shown competitive performance on the MNIST classification task against the commonly used backpropagation algorithm.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

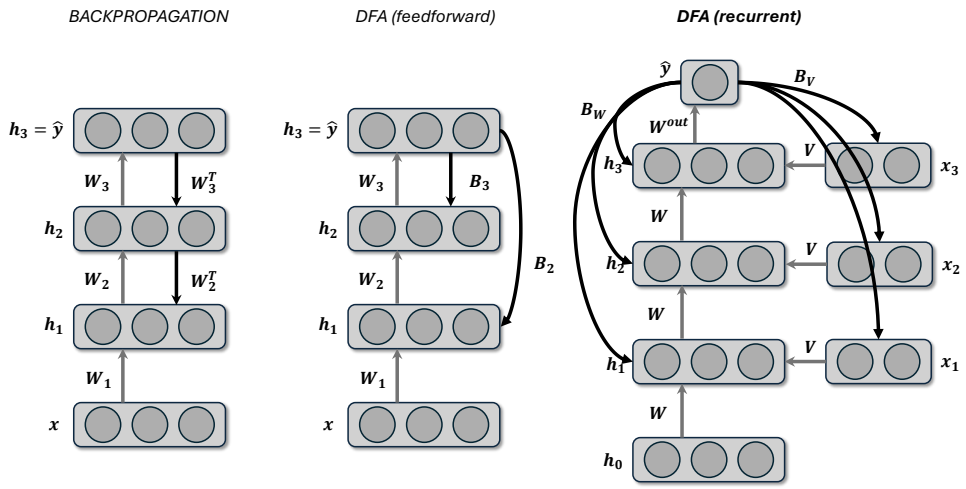


Figure 6: Comparison of backpropagation (left), DFA applied to recurrent networks (right), and DFA applied to feedforward networks (middle). In the recurrent network, the error is projected through random matrices $\boldsymbol{B_W}$ and $\boldsymbol{B_V}$, with shared weight matrices $\boldsymbol{W}$ and $\boldsymbol{V}$ across time steps (layers). In contrast, each layer of the feedforward network has a different matrix. Grey arrows represent the forward pass, and black arrows denote the update phase. While the RNN processes a sequence with multiple time steps (here, 3), the feedforward network processes a single input $\boldsymbol{x}$.

Pushing the FA idea to the extreme, Nøkland 2016 proposed DFA, where the error is randomly projected back to each layer with a direct shortcut connection.

Practical applications of DFA to RNNs have been explored in Nakajima et al. 2022. The authors performed physical deep learning with an optoelectronic recurrent neural network. However, in their pioneering work, they do not explore the DFA algorithm in the context of fully trainable RNNs, since they only provide a proof-of-concept using a reservoir computing model with untrained reservoir connections (Lukoševičius and Jaeger 2009). We investigate the potential of DFA on fully-trainable RNNs.

Han et al. 2020 investigated a DFA-inspired algorithm for RNNs. However, their version of DFA is restricted and cannot be applied to any recurrent or gated architecture, like our approach. First, they implement an upper triangular modular structure. Second, they use random projections as powers of the same matrix, which effectively resembles an FA algorithm applied to RNNs rather than a DFA algorithm for RNNs. Overall, our approach stems directly from DFA and closely follows its assumptions without requiring any customization, thus remaining more general and targeting any recurrent model.

4.2. DFA for feedforward networks

We first introduce DFA for feedforward neural networks (Figure 6, middle), to prepare the notation and set the stage for its extension to recurrent neural networks. Consider a fully-connected, feedforward neural network with an arbitrary number of L layers (including input and output layers), input size I, hidden size H and output size O. Each layer l computes its preactivation $\boldsymbol{a_l}$ through a linear projection.

$$\boldsymbol{a_l} = \boldsymbol{W_l u_l} + \boldsymbol{b_l}, \tag{20}$$

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

here $\boldsymbol{W_l} \in \mathbb{R}^{H\times I}, \mathbb{R}^{H\times H}, \mathbb{R}^{O\times H}$ is the weight matrix for the input, hidden and output layers, respectively. Similarly, $\boldsymbol{b_l} \in \mathbb{R}^H,\ l < L$ is the bias vector for the input and hidden layer and $\boldsymbol{b_L} \in \mathbb{R}^O$ is the bias vector for the output layer. The input $\boldsymbol{u_l}$ corresponds to the data sample $\boldsymbol{x}$ for the input layer ($\boldsymbol{u_1} \in \mathbb{R}^I$) and to the output of the previous layer for all other layers ($\boldsymbol{u_l} \in \mathbb{R}^H, l > 1$). The preactivation at each layer is passed through an element-wise nonlinear function σ (e.g., hyperbolic tangent) to generate the layer's activation $\boldsymbol{h_l}$. The output of the network $\hat{\boldsymbol{y}}$ is read out from the last layer.

$$\boldsymbol{h_l} = \sigma(\boldsymbol{a_l}) \tag{21}$$
$$\hat{\boldsymbol{y}} = \boldsymbol{h_L} \tag{22}$$

For each input example $\boldsymbol{x}$, the loss function $J(\hat{\boldsymbol{y}}, \boldsymbol{y})$ (e.g., cross-entropy or mean-squared error) measures the error between the output and the target prediction $\boldsymbol{y}$ associated with the example $\boldsymbol{x}$.

Updating the last layer's parameters $\boldsymbol{W_L}, \boldsymbol{b_L}$ via gradient descent is straightforward as there is a direct dependency between $\hat{\boldsymbol{y}}$ and the loss function J. For the cross-entropy or the mean-squared error loss.

$$\boldsymbol{e} = \frac{\partial J}{\partial \boldsymbol{a_L}} = \hat{\boldsymbol{y}} - \boldsymbol{y} \tag{23}$$

Therefore, $\boldsymbol{e}$ can be directly used to update $\boldsymbol{W_L}$ and $\boldsymbol{b_L}$.

$$\boldsymbol{W_L} \leftarrow \boldsymbol{W_L} - \eta \boldsymbol{e} \boldsymbol{h_{L-1}^T} \tag{24}$$
$$\boldsymbol{b_L} \leftarrow \boldsymbol{b_L} - \eta \boldsymbol{e}, \tag{25}$$

where η is the learning rate. The update of the last layer's parameters is the same for both backpropagation and DFA.

For the hidden layers, backpropagation computes the update by propagating the error signal $\boldsymbol{e}$ sequentially to lower layers (Figure 6, left). For any hidden layer, we have

$$\boldsymbol{W_l} \leftarrow \boldsymbol{W_l} - \eta(\ (\boldsymbol{W_{l+1}^T} \boldsymbol{\delta a_{l+1}} \odot \sigma'(\boldsymbol{a_l}))\ \boldsymbol{u_l^T}), \tag{26}$$

where $\odot$ denotes element-wise multiplication and $\boldsymbol{\delta a_{l+1}}$ is the error signal coming from *the layer above*. This last term requires the error to be computed sequentially one layer at a time. This dependency prevents updating all layers in parallel.

DFA removes this limitation by projecting the error $\boldsymbol{e}$ *directly* to all layers, through a random matrix $\boldsymbol{B} \in \mathbb{R}^{H\times O}$. $\boldsymbol{B}$ can also be different for each layer. Crucially, the matrix $\boldsymbol{B}$ is kept fixed and only governs the weights update. It does not take any part in the forward phase.

DFA updates each hidden layer via

$$\boldsymbol{W_l} \leftarrow \boldsymbol{W_l} - \eta(\ (\boldsymbol{Be} \odot \sigma'(\boldsymbol{a_l}))\ u_l^T), \tag{27}$$
$$\boldsymbol{b_l} \leftarrow \boldsymbol{b_l} - \eta(\ \boldsymbol{Be} \odot \sigma'(\boldsymbol{a_l})\ ). \tag{28}$$

These updates can be applied to each layer independently, thus enabling embarrassingly parallel computation for all layers.

DFA also removes the weight alignment issue, as the update circuit uses random connections instead of connections that always need to be synchronized with the forward circuit, like in backpropagation.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

4.3. DFA for recurrent networks

We develop a version of DFA that is compatible with RNNs for sequential data processing (Figure 6, right). We closely follow the DFA approach devised for feedforward networks and we extend it to the recurrent case. Each example x is a sequence of T input vectors:

$$\boldsymbol{x} = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_T}), \tag{29}$$

where $\boldsymbol{x_i} \in \mathbb{R}^I$. We consider the sequence classification task where each sequence $\boldsymbol{x}$ is associated with a target class y. The RNN keeps an internal hidden state $\boldsymbol{h} \in \mathbb{R}^H$ which is updated at each time step. We first focus on the "Vanilla" RNN (Elman 1990), whose state update of reads:

$$\boldsymbol{h_{t+1}} = \sigma(\boldsymbol{W h_t} + \boldsymbol{V x_{t+1}} + \boldsymbol{b}), \tag{30}$$

where $\boldsymbol{V} \in \mathbb{R}^{H \times I}$ is the input-to-hidden matrix and we call $\boldsymbol{a_t}$ (pre-activations at time t) the terms inside σ. In RNNs, the same layer is applied to all time steps (weight sharing). The output $\hat{\boldsymbol{y}}$ of the RNN is computed from the hidden state:

$$\hat{\boldsymbol{y}} = \sigma(\boldsymbol{W}^{\text{out}} \boldsymbol{h_t} + \boldsymbol{b}^{\text{out}}), \tag{31}$$

where $\boldsymbol{W}^{\text{out}} \in \mathbb{R}^{O \times H}$ and $\boldsymbol{b}^{\text{out}} \in \mathbb{R}^O$. The nonlinear function σ can be different from the one used in the hidden layers. For sequence classification tasks the output is computed at the end of the input sequence from $\boldsymbol{h_L}$.

Due to the weight sharing, the forward pass of an RNN can be interpreted as the unrolling of the state update function over time. At each time step, the matrix $\boldsymbol{W}$ and $\boldsymbol{V}$ (and the bias as well) are used to compute the next hidden state, much like the matrix $\boldsymbol{W_l}$ is used to compute the layer's output in a feedforward network. The backpropagation algorithm applied to RNNs (BPTT) updates the hidden-to-hidden weight $\boldsymbol{W}$ via

$$\nabla_{\boldsymbol{W}} J(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \frac{\partial J}{\partial \hat{\boldsymbol{y}}} \sum_{t=1}^{T} \frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{h_t}} \frac{\partial \boldsymbol{h_t}}{\partial \boldsymbol{W}} \tag{32}$$

The term $\frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{h_t}}$ hides a dependency between hidden states $\prod_{j=1}^{t-1} \frac{\partial \boldsymbol{h_{j+1}}}{\partial \boldsymbol{h_j}}$ which is due to the sequential propagation of the error over the time steps.

Our DFA-based algorithm for RNN removes this propagation and updates $\boldsymbol{W}$ by computing the term $\frac{\partial J}{\partial \hat{\boldsymbol{y}}} \sum_{t=1}^{T} \frac{\partial \boldsymbol{h_t}}{\partial \boldsymbol{W}}$. The error signal $\boldsymbol{e}$ is projected via a random matrix $\boldsymbol{B}$, randomly initialized and kept fixed.

The equations for the update of $\boldsymbol{W}$ and $\boldsymbol{V}$ via DFA read:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \sum_{t=1}^{T} (\, \boldsymbol{Be} \odot \sigma'(\boldsymbol{a_t}) \,)\, \boldsymbol{h_{t-1}^T}, \tag{33}$$

$$\boldsymbol{V} \leftarrow \boldsymbol{V} - \eta \sum_{t=1}^{T} (\, \boldsymbol{Be} \odot \sigma'(\boldsymbol{a_t}) \,)\, \boldsymbol{x_t^T} \tag{34}$$

The bias is updated by omitting the outer product.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Table 6: Summary of datasets statistics and average test accuracy and standard deviation over 5 repetitions for all datasets and models.

	Strawberry	LIBRAS	ECG200	Row-MNIST
Input size	1	2	1	28
Number of classes	2	15	2	10
Sequence length	235	90	96	28
Dataset size	983	360	200	70000
DFA GRU	79.73 $\pm$ 1.23	67.50 $\pm$ 3.68	80.6 $\pm$ 2.25	72.49 $\pm$ 1.1
BPTT GRU	92.05 $\pm$ 2.54	80.83 $\pm$ 9.19	82.10 $\pm$ 1.14	99.23 $\pm$ 0.03
DFA RNN	67.84 $\pm$ 2.66	47.92 $\pm$ 3.3	78.2 $\pm$ 1.47	87.48 $\pm$ 0.74
BPTT RNN	79.08 $\pm$ 4.18	54.30 $\pm$ 18.32	83.30 $\pm$ 2.1	96.69 $\pm$ 0.24

DFA for gated recurrent networks. In addition to the development of DFA for "Vanilla" RNNs (Equation 30), we also developed a version of DFA for gated recurrent networks, focusing in particular on the GRU network (Cho et al. 2014; Chung et al. 2014). The state update (forward pass) for a GRU reads:

$$
\begin{aligned}
\boldsymbol{z_{t+1}} &= \text{sig}(\boldsymbol{W_z h_t} + \boldsymbol{V_z x_{t+1}} + \boldsymbol{b_z}), \\
\boldsymbol{r_{t+1}} &= \text{sig}(\boldsymbol{W_r h_t} + \boldsymbol{V_r x_{t+1}} + \boldsymbol{b_r}), \\
\boldsymbol{c_{t+1}} &= \tanh(\boldsymbol{W_c}(\boldsymbol{h_t} \odot \boldsymbol{r_{t+1}}) + \boldsymbol{V_c x_{t+1}} + \boldsymbol{b_c}), \\
\boldsymbol{h_{t+1}} &= (1 - \boldsymbol{z_{t+1}}) \odot \boldsymbol{c_{t+1}} + \boldsymbol{z_{t+1}} \odot \boldsymbol{h_t},
\end{aligned}
$$

where *tanh* and *sig* are the hyperbolic tangent and sigmoid functions, respectively. Our DFA update for all parameters of the GRU is provided in Appendix E. The output $\hat{\boldsymbol{y}}$ of the network is computed from the hidden state $\boldsymbol{h_t}$ as previously discussed.

4.4. Experiments

We implemented all our experiments in PyTorch (Paszke et al. 2019). Although DFA does not compute a true gradient, we filled the "grad" attribute of each weight tensor with the DFA update. This enabled us to use any PyTorch optimizer to apply the update. We used the Adam optimizer for all experiments.

We assessed the performance of DFA against BPTT on the aforementioned "Vanilla" RNN and GRU. We report the average test accuracy and standard deviation computed over 5 runs. Table 6 reports a summary of the time series datasets statistics. We considered 4 different datasets:

1. *Libras*[4] Dias Daniel and Helton 2009 contains 15 classes associated with a different

[4]LIBRAS is the acronym of the Portuguese name "Lingua BRAsileira de Sinais", is the official Brazilian sign language.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

hand movement type. The hand movement is represented as a bi-dimensional curve performed by the hand in a given period of time;

2. *Row-MNIST* (Deng 2012): each image of the MNIST dataset is presented to the recurrent model one row at a time;
3. *ECG200* (Olszewski, Maxion, and Siewiorek 2001): where each time series traces the electrical activity of a subject recorded during one heartbeat. The task is a binary classification prediction between a normal heartbeat and one highlighting a Myocardial Infarction;
4. *Strawberry* (K. Kemsley n.d.) consists in classifying food spectrographs, a task with applications in food safety and quality assurance. The classes are strawberry (authentic samples) and non-strawberry (adulterated strawberries and other fruits).

The datasets are divided into train, validation and test sets according to the proportions 60%-20%-20%. The hyperparameters have been selected based on a model selection with a grid search (see Appendix F for the details).

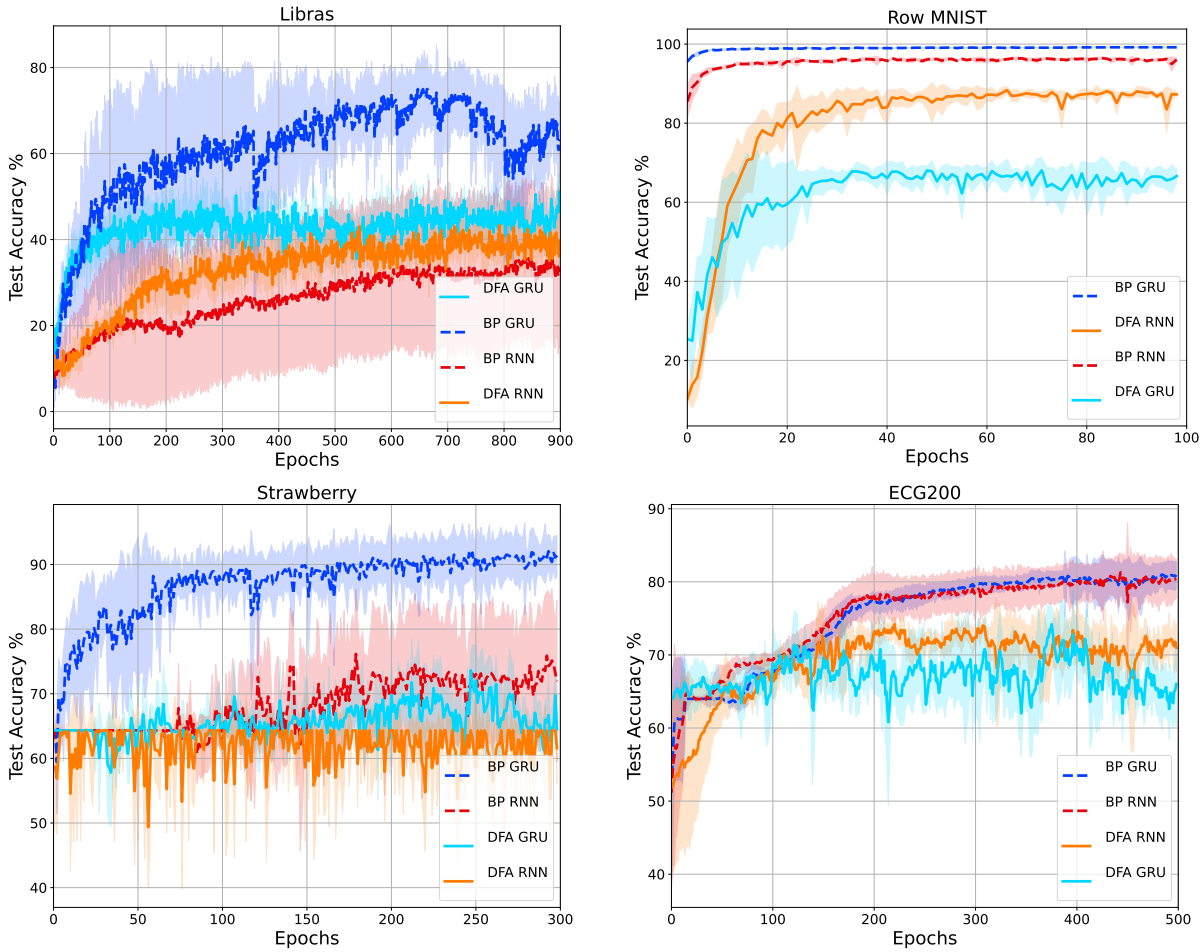


Figure 7: Results on the Libras, Row-MNIST, Strawberry and ECG-200 datasets with a "Vanilla" RNN architecture (orange and red) and with a GRU (blue and cyan). The models are trained with DFA (lighter colors, full line) and BPTT (darker colors, dashed line). Error shades denote one standard deviation computed over 5 repetitions with different seeds.

Table 6 reports the test accuracy achieved by all methods, alongside the specifics of the datasets. Overall, BPTT still outperforms DFA across most datasets. Specifically, BPTT out-

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

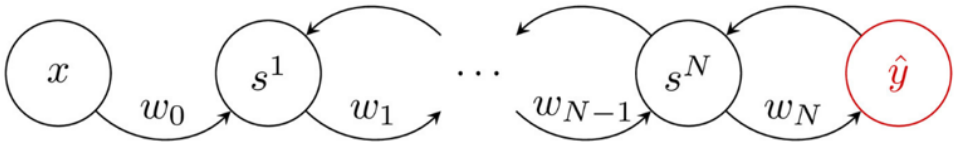


Figure 8: Equilibrium Propagation on a generic layered recurrent architecture. RON implements an oscillatory dynamics at each layer. Note that the connections work in both the forward (from the input to the output) and backward (from the output to the input) directions.

performs DFA with GRU architectures except for the ECG200 dataset, in which both learning algorithms achieve a comparable performance.

With "Vanilla" RNN architectures, BPTT outperforms DFA except for the ECG200 and the Libras datasets, where the average test accuracy of DFA (Figure 7 top-left panel, orange line) is higher than BPTT's one (red line) after the first 150 epochs. Moreover, in this dataset, DFA has the same learning slope of BPTT either with vanilla RNNs (for the first 150 epochs) or for GRUs (for the first 50 Epochs).

DFA seems to struggle with unbalanced datasets, like ECG200 and Strawberry. In the ECG dataset, which is the one with the smallest amount of data, the test accuracy of RNN with DFA is above the random performance of 12%. In the Strawberry dataset, the same model with DFA shows an accuracy which is above the random performance of only 5%. In the case of balanced datasets, RNNs trained with DFA are generally successful at learning temporal correlations.

Overall, while BPTT generally resulted in higher test accuracy, DFA demonstrated comparable performance particularly for ECG200 in both GRU and RNN models. This suggests that although DFA is less accurate overall, it may be a viable alternative in scenarios where strong parallelization combined with a physical implementation is a possibility.

There are still other aspects that require further consideration. For example, the choice of the random feedback matrix is crucial, as it affects the trajectory of the parameters during training. Moreover, different matrix structures are amenable to different implementations in neuromorphic or unconventional hardware. Crafton et al. 2019 implemented DFA for feedforward architectures on neuromorphic hardware with a sparse feedback matrix, at minimal or no performance loss.

Our algorithm can also be easily extended to deal with time series forecasting tasks, where the prediction step is taken after each time step, instead of only at the end of the input sequence. Further benchmarking of our DFA in these settings is required to understand its effectiveness.

5. Learning via Equilibrium Propagation

Equilibrium Propagation (EP) (Scellier and Bengio 2017), like Direct Feedback Alignment, is a backpropagation-free algorithm for updating the parameters of an adaptive model. EP is specifically designed for adaptive dynamical systems and, as such, it is an ideal candidate to train archetype networks.

Following Ernoult et al. 2019, we first consider a hierarchical dynamical system resembling a deep "vanilla" recurrent network. The state update for a generic hidden layer l reads:

$$\boldsymbol{s}_{\boldsymbol{t+1}}^{\boldsymbol{l}} = \sigma(\boldsymbol{W}_{\boldsymbol{b}}^{\boldsymbol{l}} \boldsymbol{s}_{\boldsymbol{t}}^{\boldsymbol{l+1}} + \boldsymbol{W}_{\boldsymbol{f}}^{\boldsymbol{l}} \boldsymbol{s}_{\boldsymbol{t}}^{\boldsymbol{l-1}}), \tag{35}$$

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

where the sets of connections $\boldsymbol{W_f}, \boldsymbol{W_b}$ governs the propagation of the network's activations in the forward (from the input to the output) and backward (from the output to the input) directions, respectively. In the case of symmetric connections one can use a single symmetric $\boldsymbol{W}$ matrix. In the case of the input layer, its state is clamped to the value of the external input $\boldsymbol{x}$. In the case of the output layer, the state update only considers the forward connections coming from the last hidden layer, thus omitting the backward term and leading to:

$$\boldsymbol{s_{t+1}^L} = \sigma(\boldsymbol{W_f^L s_t^{L-1}}). \tag{36}$$

EP requires two phases: during the first phase the input $\boldsymbol{x}$ is propagated through the network in the forward and backward directions by iteratively computing Equation 35. After a certain amount of steps $t \in [1, K]$ the first phase terminates. K is a hyper-parameter of EP. The second phase starts from the last activations computed during the first phase. In addition, the state update of the output layer is modified as:

$$\boldsymbol{s_{t+1}^L} = \sigma(\boldsymbol{W_f^L s_t^{L-1}}) + \beta \mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{s_t^L}), \tag{37}$$

where β is a scalar hyper-parameter and $\mathcal{L}$ is a loss function that computes the distance from the true target $\hat{\boldsymbol{y}}$ and the output of the network. A popular choice for the loss function is the Mean Squared Error (MSE). The second phase lasts for T steps (a hyper-parameter) and updates all the network's states.

Figure 8 summarizes the architecture we considered for the EP training algorithm.

At the end of such process, EP updates a generic parameter $\boldsymbol{W}$ of the network via:

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \frac{\eta}{\beta}\left(\frac{\partial \Phi}{\partial \boldsymbol{W}}(\boldsymbol{x}, \boldsymbol{s_*}, \boldsymbol{W}) - \frac{\partial \Phi}{\partial \boldsymbol{W}}(\boldsymbol{x}, \boldsymbol{s_*^\beta}, \boldsymbol{W})\right), \tag{38}$$

where η is the learning rate, $\boldsymbol{s_*}$ are the activations of the network after the first K steps and $\boldsymbol{s_*^\beta}$ are the activations of the network after the first $K + T$ steps. The function Φ denotes the primitive function regulating the dynamics of the network. It is defined such that $\frac{\partial \Phi}{\partial \boldsymbol{s}}(\boldsymbol{x}, \boldsymbol{s_t}, \boldsymbol{W}) = \boldsymbol{s_{t+1}}$. Usually, the nonlinear function σ is not considered by Φ. We can define the primitive function for the recurrent network of Equation 36 as: $\Phi(\boldsymbol{x}, \boldsymbol{s_t}, \boldsymbol{W}) = \sum_{l=1}^{L} \boldsymbol{s^l W_l s^{l+1}}$. We also considered the RON model (Equation 19, where $\boldsymbol{W}$ is a dense matrix instead of antisymmetric) and trained it with EP on a static input. We added the oscillatory behavior to the vanilla RNN unit of Equation 36. We kept the oscillators fixed and trained the recurrent weight matrices. This allows us to leverage the same Φ function to update the parameters, while using an oscillatory state-update function.

On both MNIST and CIFAR10 datasets, RON trained with EP matches or surpasses the performance of the vanilla RNN of Equation 36: we achieve 98% accuracy on MNIST and 47% accuracy on CIFAR10.

EP for time series processing. We adapted EP to maintain a memory of previous inputs to process time series. Figure 9 provides an overview of our approach. We consider an input time series $\boldsymbol{x}$ of T steps. Each step is processed by either a vanilla RNN or RON. For each step of the input sequence, EP performs both the first and the second phase previously describe. At the end of the second phase, EP updates the parameters of the network. The state computed at the end of the first phase are kept as the initialization for the subsequent input value from the time series. The experiments are still ongoing. To the best of our knowledge, we are the first to consider EP to learn adaptive dynamical systems for time-varying inputs.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

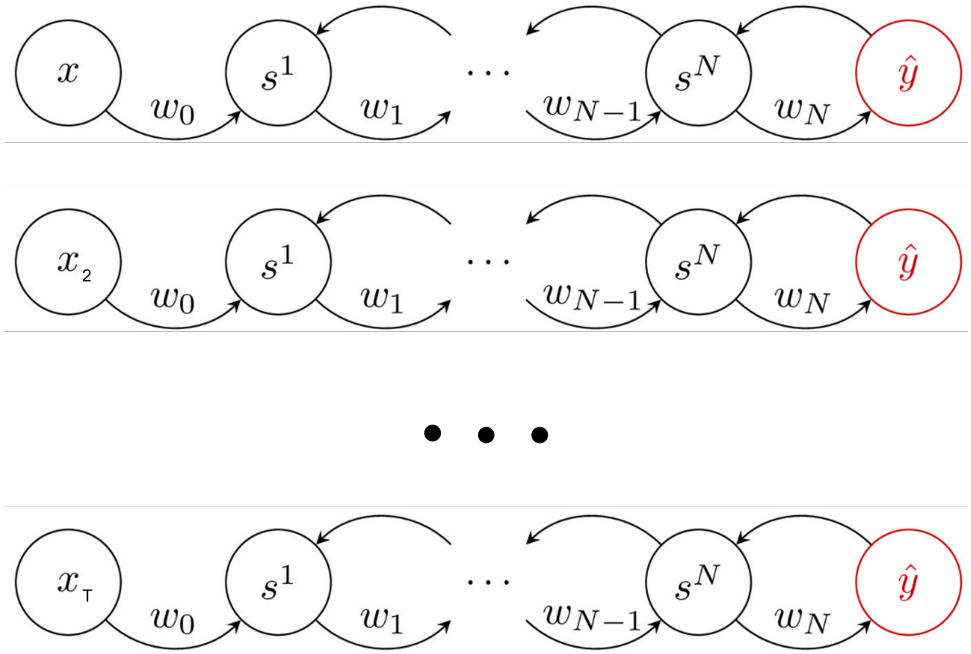


Figure 9: Equilibrium Propagation on a generic layered recurrent architecture with a time-varying input.

6. Life-long evolutionary swarms

A swarm is composed of simple agents that interact locally and give rise to a decentralized, self-organized behavior towards a common goal. Designing swarm controllers is a critical challenge, as manually defining the individual rules to achieve a desired collective behavior is often unfeasible due to the complex interactions and emergent properties of swarms. Evolutionary algorithms provide an effective solution to this challenge, bypassing the need for manual intervention and letting the environment decide which solutions are worth exploring more than others. Yet, to truly operate in complex dynamic environments, swarm systems must not only adapt to novel challenges but also retain previously acquired skills. For example, a swarm that needs to solve a foraging task may be required to fetch different types of objects and/or bring them to different locations. Changes in the task (e.g., fetching a different object) require the swarm controller to adapt. Subsequent changes in the task should not erase all the information, since previous tasks might occur again. A swarm able to preserve previous knowledge will be able to adapt faster to reoccurring tasks than a swarm that always starts from scratch.

This is precisely the focus of *lifelong learning* (Parisi et al. 2019), a research topic mainly explored within deep learning (Masana et al. 2023) and, to a lesser extent, robotics (Lesort et al. 2020). In lifelong learning a single agent, usually an artificial neural network, is trained via backpropagation on a sequence of tasks. The network has to adapt to the new task and preserve the performance on previously encountered tasks.

We merge the evolutionary and lifelong paradigms and propose a lifelong evolutionary environment (Section 6.2) for the design of swarm controllers (Section 6.3). Our objective is to enable swarms to quickly adapt to new tasks while preserving previous knowledge. We study the performance of the evolved population over the course of several hundred generations, with particular attention to the drifting points when a new task replaces the current one in the environment (Section 6.4.1). Inspired by lifelong learning, we also study the performance of the *top-performing individual* on a given task, and how much knowledge of previous tasks it

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

retains. This is useful when deploying a single evolved controller in a dynamic environment that can change at any moment. We discovered that, while the population inherently exhibits a certain degree of retention due to its diversity (Section 6.4.2), the individual catastrophically forgets previous knowledge (Section 6.4.3). Similar to regularization approaches in lifelong learning (Parisi et al. 2019; Kirkpatrick et al. 2017), we propose a regularized version of the evolution process that is able to mitigate forgetting for the top-performing individual, achieving a controller that trades-off the performance on previous tasks and adaptation to the current task.

The deep lifelong learning perspective based on a single agent may not be the only choice to build adaptive systems that retain knowledge, as we discovered that an evolved population with enough diversity is able to partially satisfy this requirement without an ad-hoc design. On the other hand, ideas from lifelong learning can help guide the evolutionary process, opening new challenges for the design of swarm controllers in dynamic environments.

6.1. Background

The evolution of swarm controllers focuses on environments where the task to solve do not change over time. The literature is currently lacking a thorough study on how evolutionary algorithms behave when the population of controllers needs not only to adapt to a given task but also to retain information from previously encountered tasks. This challenge is instead the focus of lifelong learning, and it is usually studied for single agents implemented by deep artificial neural networks.

We therefore provide a short overview on lifelong learning first, and on evolution of swarm controllers next. We devote particular attention to those aspects that will be leveraged for our lifelong evolutionary swarms framework.

Lifelong learning. Lifelong learning (Parisi et al. 2019), also called continual learning (Cossu, Bacciu, et al. 2021), designs agents that learn from a stream of observations without forgetting previous knowledge. Lifelong learning is mainly studied in the context of deep learning (Verwimp et al. 2024) and reinforcement learning (Khetarpal et al. 2022), where the learning agent is a deep artificial neural network trained with backpropagation. However, its scope is much broader, as it encompasses any environment where the task to solve is drifting over time (Giannini et al. 2024) and it is not tied to the specific learning approach or model.

Recently, lifelong learning research focused on the forgetting issue that plagues neural networks and other predictive models (French 1999): upon learning new information, the model's performance on previously observed data quickly deteriorates. For example, incrementally adding new classes in a classification problem can reduce the model's performance on previously encountered classes. Forgetting is mainly due to the network's inability to maintain a stability-plasticity trade-off (Carpenter and Grossberg 1986). Overly plastic networks tend to forget previous knowledge, while overly stable networks are unable to adapt to new tasks. To mitigate forgetting, a plethora of different solutions have been developed (Masana et al. 2023). One of the most popular approaches to combat forgetting is Elastic Weight Consolidation (EWC) (Kirkpatrick et al. 2017) and variants thereof (Zenke, Poole, and Ganguli 2017; Chaudhry et al. 2018). The idea behind these *importance-based regularization approaches* (Parisi et al. 2019) is to limit the plasticity of synaptic connections that are deemed important for previous tasks. The plasticity is computed with respect to a previous version of the same model that performed well on previous tasks. As a consequence, the network will be forced to leverage the remaining connections for the adaptation to new incoming tasks, with an

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

improved stability of previous performance. Computing the importance of a given synaptic connection is at the heart of importance-based regularization strategies. Often times, the resulting performance represents a trade-off between the ideal performance on new tasks and the ideal performance on previous tasks. We exploit this idea and adapt it in our lifelong evolutionary swarms framework to mitigate forgetting for the top-performing individual of a population.

Swarm robotics Swarm robotics takes inspiration from swarms in nature with collective behaviour emerging through local interactions between agents and with the environment (Şahin and A. Winfield 2008). A fundamental issue is the design of controllers such that a desired collective behaviour emerges, common approaches include bioinspiration, evolution, hand design, reverse engineering (Reynolds 1987; Hauert, Zufferey, and Floreano 2009; A. F. Winfield 2009; Francesca and Birattari 2016; Jones, A. F. Winfield, et al. 2019). Making swarms adaptive to change is related to lifelong learning (Yao, Marchal, and Van de Peer 2014; Castello et al. 2016). See Bredeche, Haasdijk, and Prieto 2018; Birattari et al. 2019; Schranz et al. 2020; Dorigo, Theraulaz, and Trianni 2021 for recent reviews. Much work has focussed on neuroevolution of controllers; one state-of-the-art approach is NeuroEvolution of Augmenting Topologies (NEAT) (Kenneth O. Stanley and Risto Miikkulainen 2002), which we use here. NEAT not only searches for optimal neural network weights but also evolves network topologies in search of the best *minimal* architecture. NEAT has been effectively utilized to solve highly complex problems, such as double pole balancing, where it outperforms several methods that rely on fixed topologies (Kenneth Owen Stanley and R. P. Miikkulainen 2004). The algorithm's superior performance can be attributed to three key features: the use of historical markers to enable meaningful crossover between different topologies, a niching mechanism (speciation), and the gradual evolution of topologies starting from simple initial structures (complexification).

In swarm robotics, NEAT has been applied to evolve neural networks for autonomous foraging tasks, as demonstrated by NeatFA (Ericksen, M. Moses, and Forrest 2017). This approach achieved performance comparable to or surpassing established algorithms such as the Central Place Foraging Algorithm (Hecker and M. E. Moses 2015) (CPFA) and the Distributed Deterministic Spiral Algorithm (DDSA) (Fricke et al. 2016). NEAT has also been used in distributed online learning with swarms (odNEAT) (Silva et al. 2015). odNEAT operates across multiple robots which have to solve the same task, either individually or collectively. A significant aspect of odNEAT is its dynamic population management, where each individual maintains an internal set of genomes, including current and previously successful controllers. odNEAT provides results comparable to centralised approaches (K. Stanley, Bryant, and R. Miikkulainen 2005).

Lifelong neuroevolution In the context of lifelong evolution of neural networks, (Ellefsen, Mouret, and Clune 2015) investigates how evolving modular neural networks can mitigate catastrophic forgetting by reducing interference between tasks. The setup involves an abstract environment where organisms evolve to maximize fitness by learning to consume nutritious food and avoid poisonous food, with abrupt seasonal changes introducing new food sources. By encouraging modularity through a connection cost mechanism, networks evolve to separate functionality into distinct modules, enabling selective learning. This modularity improves performance by allowing agents to learn new skills faster while retaining old ones. The study highlights that modularity not only enhances learning dynamics but may also reflect an evolutionary advantage observed in natural animal brains to combat catastrophic forgetting.

The study in Kashtan, Noor, and Alon 2007 explored how modularly varying goals (MVG) can accelerate evolutionary processes in computer simulations. Leveraging on genetic algorithms, the authors investigated how populations of networks evolve under temporally changing goals,

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

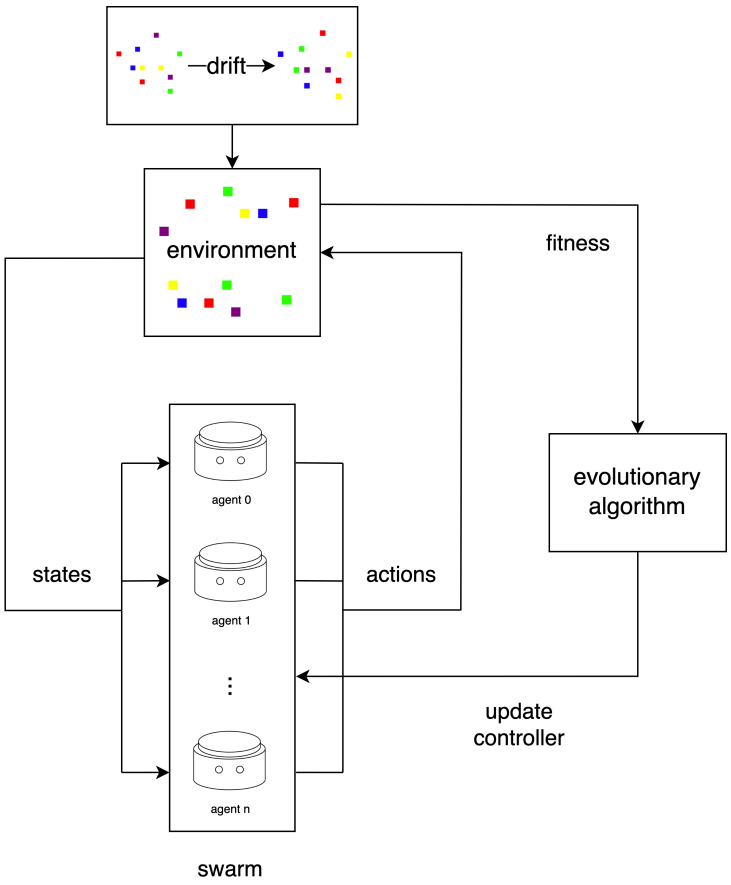


Figure 10: Overview of the lifelong evolutionary swarms framework. The swarm, composed of multiple agents, interacts with the environment through actions based on local sensor inputs (environment state) processed by an identical internal controller. Each agent keeps a copy of the controller. The sequence of actions of an agent determines the fitness of the controller. The evolutionary algorithm updates the controller based on their fitness. Periodically, the dynamic environment changes the underlying task which requires the swarm to adapt to novel conditions without forgetting the previous knowledge.

where each new goal shared subproblems with ones. The tasks involved combinatorial logic problems, such as solving specific input-output relationships, which grew in complexity with modular goal changes. Their results showed that MVG significantly reduced the number of generations required to achieve a given goal compared to fixed-goal evolution.

To the best of our knowledge, we are the first to focus on lifelong neuroevolution of swarm controllers.

6.2. An Environment for the Lifelong Evolution of Swarms

To study and address the challenges of i) quick adaptation in the presence of task drifts and ii) retention of previous knowledge in swarm controllers, we present our lifelong evolutionary framework (Figure 10). This section only assumes that the swarm controller is optimized with an evolutionary algorithm that keeps a population P of controllers (individuals) $\pi \in P$. We defer the discussion about the evolutionary algorithm we adopted to Section 6.3.

Each agent in the swarm perceives the environment through their sensors. The perceived

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

state is local to the agent. The same controller is deployed across all agents, forming a homogeneous swarm. It processes the local environment state, along with any additional external information, to determine each agent's next action. We define an *episode-based* environment, where at the start of each episode the agents are positioned in the environment and they move and operate for a fixed amount of K steps, at the end of which the episode terminates. A step involves each robot sensing the environment, providing the observations as input to the controller, generating output commands, sending them to the actuators, and moving to a new position. The swarm's objective is to solve a task, implicitly defined by a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which takes an environment state $\mathbf{s} \in \mathcal{S}$, an action $\mathbf{a} \in \mathcal{A}$ and returns a scalar reward. The rewards are summed across all K steps. Notice that the proposed setup is not tied to a specific task, but it can be used with any reasonable reward function.

At each generation of the evolutionary algorithm, each controller is evaluated across N randomly generated environments with varying initial states, but with the same reward function. The final fitness score is computed as the average total reward over the evaluation environments. Formally, the fitness for task t of a controller π[5] is:

$$f_t(\pi) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \mathcal{R}_t(\mathbf{a}_k = \pi(\mathbf{s}_k), \mathbf{s}_k). \tag{39}$$

The controller receives the current state $\boldsymbol{s}_k$ and returns the action $\boldsymbol{a}_k$. The reward function, as the fitness, explicitly depends on the task t (Section 6.2.1 provides our choice for the reward function). The fitness is used by the evolutionary algorithm to update the controllers.

The key innovation of our framework is the introduction of dynamic target objectives that change (drift) over the population's lifetime. This sudden change mimics real-world environmental shifts where the relationship between actions and rewards is redefined. This corresponds to a change in the mapping implemented by $\mathcal{R}$. As a result, the top-performing controller will see a decrease of its fitness, as it faces a new objective. We monitor the performance on previous tasks by adopting the same evaluation protocol mentioned above. For example, when evolving a controller on the second task, we measure its average fitness over N randomly-generated environment according to the reward function of the second task (*adaptation*), and on another N environments according to the reward function of the first task (*retention*).

We formally define our evaluation metrics. We denote the retention of a (previous) task $t-1$ for a controller π in a population P as $f_{t-1}(\pi)$. Among all individuals in a population P_t evolved for the *current* task t, we are particularly interested in the one with the highest current (C) fitness:

$$C(P_t) = \max_{\pi \in P_t} f_t(\pi). \tag{40}$$

This metrics quantifies the effectiveness of the evolutionary algorithm to find good controllers for the current task t.

We can consider retention at two levels:

- *Population Level*: we examine whether some individuals in the population retain knowledge from previous tasks. Such individuals are crucial as they can guide the evolutionary process to quickly re-adapt when re-encountering a previously faced task. A diverse population is expected to exhibit varying levels of retention across individuals. To capture

[5]Note that π defines the controller phenotype (a function mapping states to actions). The evolutionary algorithm searches for fit phenotypes by acting on the corresponding genotypes.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

this, we evaluate each individual on the previous task and identify the one that retains the highest performance (R^{pop}), i.e., the individual with the largest retention:

$$R^{pop}_{t-1}(P_t) = \max_{\pi \in P_t} f_{t-1}(\pi). \tag{41}$$

This metric is aligned with a multi-agent system, where we always consider that there is a population of solutions, and not a single agent to be optimized.

- *Individual Level*: we measure the retention (R^{top}) of the best individual for the current task on a previous task. This is particularly relevant when the population can no longer evolve, such as during deployment, where a single controller is needed for inference. This is also a more challenging objective, as the best-performing individual for the current task may evolve specific skills that are useful for that task, and less useful for others:

$$R^{top}_{t-1}(P_t) = f_{t-1}\left(\arg\max_{\pi \in P_t} f_t(\pi)\right). \tag{42}$$

This view is aligned with the lifelong learning approach, where a single agent is optimized on a stream of tasks.

The main reason we are interested in assessing the retention of previous knowledge is that previous tasks may reappear in the future. This is a well-known idea in lifelong learning, where an environment with repeating tasks can be exploited by the agent to improve its performance (Hemati et al. 2023). Interestingly, a large body of works in lifelong learning focuses on environments that never repeat previous tasks, although the agent is still required to retain knowledge about them; a case which is not that common in the real-world (Cossu, Graffieti, et al. 2022).

We define *forgetting* F_{t-1} for task $t-1$ as the difference between the best performance previously achieved on that task and the retention after evolving for the subsequent task. Formally:

$$F^{pop/top}_{t-1} = C(P_{t-1}) - R^{pop/top}_{t-1}(P_t). \tag{43}$$

Note that forgetting compares the performance on the *same task* (same reward function) of *two different populations*. Although forgetting can be computed for the population at any generation, it is most insightful when evaluated at the last generation of each task.

To experiment with our lifelong evolutionary framework, we chose *foraging* tasks as a well-established benchmark in swarm robotics. Foraging requires the swarm to explore the environment, identify target resources and transport them to a pre-defined drop zone. Foraging is a useful benchmark for lifelong learning, as it can easily be extended to include changes in the reward function. Our implementation requires the swarm to locate a set of boxes marked by a target color. After a certain time, the target color changes and the swarm has to fetch the new boxes instead.

6.2.1. Environment Setup

We implemented the foraging environment as a 2D continuous-space simulation using the Gymnasium[6] library in Python (Towers et al. 2024). The environment consists of a 5-meter $\times$ 5-meter arena populated with swarm of agents and a set of color-coded boxes scattered across the entire environment. The drop zone is positioned along the arena's upper edge. Once an

[6] `https://gymnasium.farama.org`

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Table 7: The input and output nodes for the swarm's neural controller. c is the number of possible colors available in the arena and n is the number of neighbors perceived.

Input	**Nodes**	**Value**
neighbor type	$(c+3) \times n$	$\{$*none, wall, agent, color*$_{1,...,c}\}$
neighbor distance	n	[0, 1]
neighbor direction	$2 \times n$	[0, 1]
heading	2	[0, 1]
carrying box	$c+1$	$\{$*none, color*$_1$*, ..., color*$_c\}$
target color	c	$\{$*color*$_1$*, ..., color*$_c\}$
Output	**Nodes**	**Value**
wheel velocities	3	$[-v_{max}, v_{max}]$

agent drops a box, it goes on to fetch another one. The simulation expires after a fixed amount of time.

Each robot and box is represented as a point in the arena, with a diameter of 250 mm. Robots pick up boxes by coming into contact with them, defined as the robot's center being within a 125 mm radius of the box's center. Similarly, they drop boxes when positioned near the drop zone. Robots are equipped with omnidirectional wheels, enabling movement in any direction at a maximum speed of 50 cm/s.

Robots operate solely on individual sensor input to replicate realistic swarm scenarios. Robots lack knowledge of their global position, the location of boxes, or the drop zone.

To emulate the capabilities of real-world DOTS robots (Jones, Milner, et al. 2022), which serve as the basis for our study, it is assumed that each robot can perceive the type, distance, and orientation of entities (including other robots, walls, and boxes) within a 1m range. Robots can also identify whether they are carrying a box, and determine their heading direction using an onboard compass. The target color can be provided as input to the controller if needed.

For the experiments, the arena is populated with a swarm of five robots and 20 boxes of various colors, with half matching the target color and the other half having a different one. Each simulation episode includes $K = 500$ steps, with each step representing 0.1 seconds. We used $N = 10$ evaluation environments.

Positive individual behaviors, such as picking up a correctly colored box, earn one point, while delivering it to the drop zone earns two points. Conversely, picking up a wrongly colored box results in a penalty of -1 point. Given $\mathbf{s}, \mathbf{a}$ as two vectors defining the perceived state and resulting action of each agent in the swarm, we define our reward function for task t as:

$$\mathcal{R}_t(\mathbf{s}, \mathbf{a}) = \sum_{(s_i, a_i) \in (\mathbf{s}, \mathbf{a})} \begin{cases} +1, & \text{picking up a target } t \text{ box,} \\ +2, & \text{delivering a target } t \text{ box,} \\ -1, & \text{picking up a non-target } t \text{ box,} \\ 0 & \text{else.} \end{cases}$$

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

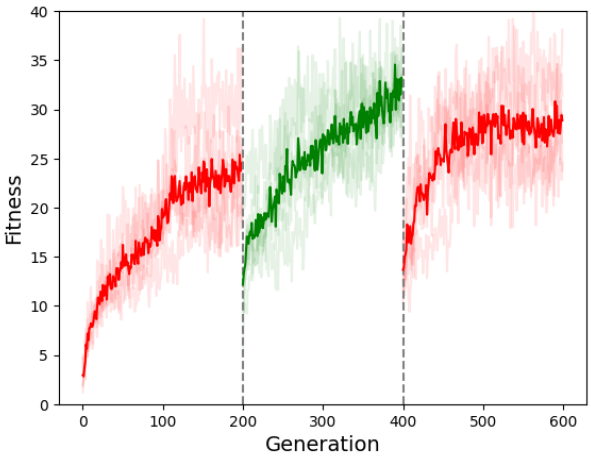


Figure 11: Fitness evolution of the best individual in the population for each generation and across three tasks: red task (generations 0–200), blue task (generations 200–400), and a return to the red task (generations 400–600). Line colors correspond to the task, and vertical dashed lines mark the transitions between tasks. The rapid recovery of fitness after task switches highlights the system's adaptability and transfer of knowledge between related tasks.

6.3. Lifelong Neuroevolution of Swarm Controllers

The swarm's behavior is governed by an artificial neural network controller evolved using NEAT (Kenneth O. Stanley and Risto Miikkulainen 2002). NEAT evolves both the network's topology and its weights. Network inputs consist of the robot's local environmental observations, while the outputs correspond to velocity commands for the robot's three wheels. Before being processed by the neural controller, observations are preprocessed. Categorical variables, such as the type of perceived entities or the type of box being carried, are one-hot encoded. Continuous variables are normalized to a range of 0 to 1, and angular directions are transformed into sine and cosine components, ensuring smooth representation and continuity between angles like 1° and 360°. A detailed breakdown of inputs and outputs is provided in Table 7.

Mitigating forgetting via genetic distance regularization As we will see in Section 6.4.3, the best performing individual on a given task is subject to forgetting of previous knowledge. To mitigate this phenomenon we introduce a *genetic distance* regularization method, inspired by Elastic Weight Consolidation (EWC) (Kirkpatrick et al. 2017) from lifelong learning. Our approach adds a penalty term to the fitness function, like EWC adds a penalty term to the loss function. EWC mitigates forgetting by pulling *important* weights of the current network towards reference values that were effective for previous tasks. While originally designed for deep learning, we adapted the same principle for the evolutionary algorithm used in our framework by i) selecting important connections, ii) choosing a reference model, iii) pulling existing weights toward their reference.

i) EWC uses the Fisher Information Matrix to select the important connections. Since we are dealing with a gradient-free evolutionary approach, the Fisher Information is not available. Our approach starts with a small network without hidden layers and evolve it using NEAT. This allows the architecture to expand dynamically and to add or remove connections and nodes. Therefore, we assume that all existing connections are important, as unnecessary ones would likely be removed during the evolutionary process.
ii) It is straightforward for EWC to choose the reference model, as it trains a single network.

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

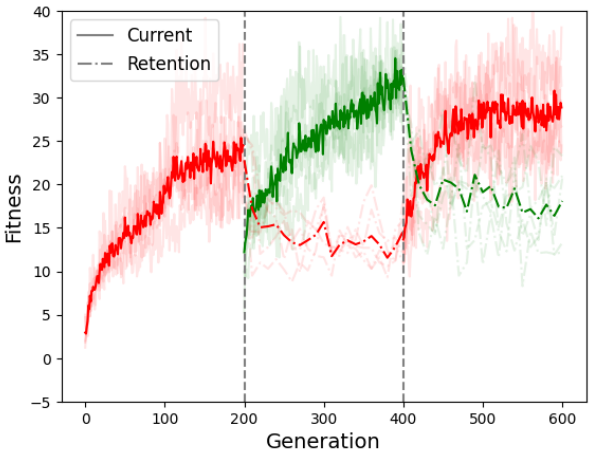


Figure 12: Current performance (solid lines) and retention at the population level (dashed lines) across task drift (line colors correspond to task colors). Population naturally preserves past knowledge while adapting to new objectives.

Evolutionary techniques like NEAT, instead, maintain a population of solutions and thus require a selection process to determine the best reference model (e.g., the simplest, the fittest, or the network which changed the least). We select the individual with the highest *regularized fitness* (Equation 45 below) as the reference model. Moreover, the network's architecture in NEAT changes across generations. Therefore, we need to map the architecture of the current individual to the reference network's architecture. Given two controller's genotypes $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$, we use the NEAT routine δ to compute their genetic distance:

$$\delta(\boldsymbol{x_1}, \boldsymbol{x_2}) = \frac{c_1 E(\boldsymbol{x_1}, \boldsymbol{x_2})}{\max\{|\boldsymbol{x_1}|, |\boldsymbol{x_2}|\}} + \frac{c_2 D(\boldsymbol{x_1}, \boldsymbol{x_2})}{\max\{|\boldsymbol{x_1}|, |\boldsymbol{x_2}|\}} + c_3 \cdot \overline{W}(\boldsymbol{x_1}, \boldsymbol{x_2}). \tag{44}$$

The function is a linear combination of the number of excess genes E (present after the last matching gene), disjoint genes D (non-matching genes between aligned matching genes), and the average weight difference of matching genes $\overline{W}$ (including disabled ones), weighted by coefficients c_1, c_2, c_3 and normalized based on the largest genome size.
iii) Our approach computes the *regularised fitness* as:

$$f_t^{gd}(\pi) = f_t(\pi) - \lambda\delta(x_{\pi^*}, x_\pi), \tag{45}$$

x_π denotes the genotype of an individual π from the current population, and x_{π^*} is the genotype of the reference model π^* from the previous task. The term λ is a weighting coefficient that controls the influence of the genetic distance penalty.
As with other regularization techniques in lifelong learning, we expect a trade-off between the performance on the current task and the performance on previous tasks. A high regularization will prevent the population to adapt, while a low regularization will not preserve previous knowledge.

6.4. Experiments

We evaluate our lifelong evolutionary framework for swarms according to i) their ability to adapt to novel tasks, including transferring knowledge from previous tasks to increase the performance on the current one; ii) their ability to retain knowledge from previous tasks; iii) the ability of the top-performing individual to mitigate forgetting with our genetic distance regularization.

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

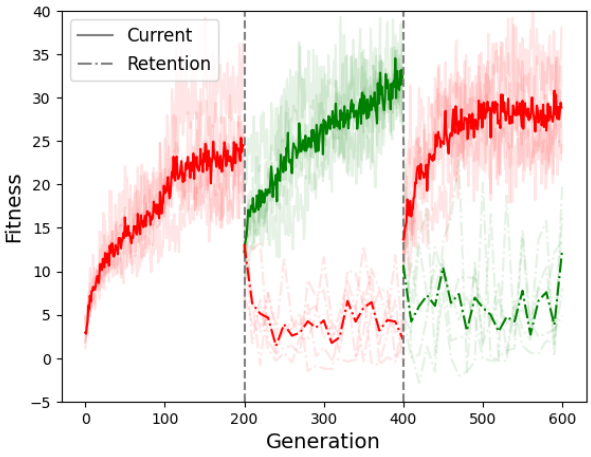


Figure 13: Current performance (solid lines) and retention at the individual level (dashed lines) across task drift (line colors correspond to task colors). Retention fitness demonstrates catastrophic forgetting, as the best-performing individual on the current task fails to retain knowledge of prior tasks.

We present results averaged over five random seeds that control the initial network configuration, the stochastic genetic operations (recombination and mutation), and the initial agents and boxes location. This means that runs that achieve a similar fitness values may end up with very different networks, potentially belonging to different NEAT species.

6.4.1. Quick Adaptability

We consider a dynamic environment where the task switches from retrieving red boxes to retrieving green boxes and then it switches back to the red boxes. Upon task change, the swarm should be able to quickly adapt to the new task (after a short transient). We also expect the swarm to reuse previous knowlegde. This means that, whenever a previous task reappears, the swarm's performance should be higher than the one of a randomly-initialized swarm controller.

Figure 11 shows the fitness of the top individual for each generation across the three tasks. We observe that at generation 0 the swarm is unable to retrieve boxes (fitness around 0), as the controller is still a randomly initialized one. As the evolution progresses the fitness steadily increases up until 25 at generation 200, when the task drifts to the green boxes. The fitness then suddenly decreases as the previously evolved controller still retrieves red boxes. However, it does not drop to 0, highlighting that the swarm can indeed exploit knowledge from the previous task. Morever, the swarm adapts much quicker than before, reaching the same fitness of the red task in less than half of the generations previously required. This is further evidence that prior learning on the red task has facilitated faster adaptation to the new objective.

The same rapid recovery is also observed after generation 400, when the red task reoccurs. Within the first 50 generations, the previous performance on the red task achieved at generation 200 is fully restored. The lifelong evolutionary process is able to leverage prior knowledge in order to speed up convergence on new tasks, requiring significantly fewer generations. Compared to a static evolution setup, where each task is evolved separately and by starting from a randomly-initialized population, lifelong evolution requires less resources and can effectively transfer knowledge across similar tasks.

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

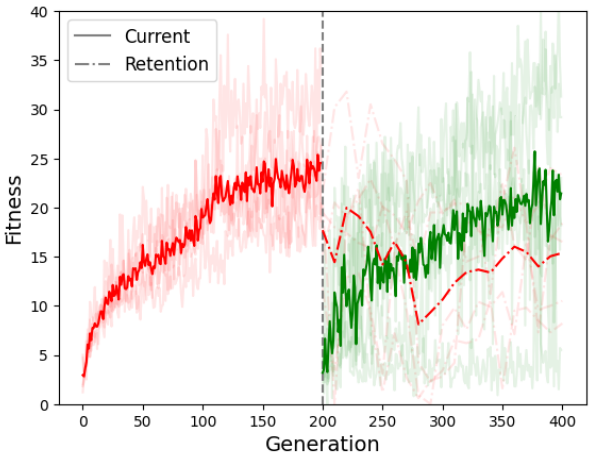


Figure 14: Current performance (solid lines) and retention at the individual level (dashed lines) for the red and green tasks when applying a fixed regularization coefficient, $\lambda = 11$. Retention is higher than without regularization (mitigating forgetting), though at the cost of a reduced performance on the green task.

6.4.2. Forgetting in the population

In the spirit of lifelong learning, we measure the ability of the evolved population to mitigate forgetting by retaining previous knowledge. We use the same dynamic environment of Section 6.4.1. Every 10 generations we evaluate the fitness on the previous task of each individual in the population. Figure 12 shows the fitness of the best-performing individual at each generation according to Equation 41. We explicitly plot the retention curve, while the forgetting is obtained by comparing the dashed red curve at generation 400 (performance on previous red task when evolving on the green task) with the solid red curve at generation 200 (performance on the red task at the end of evolution on the red task).

The red retention curve shows that at each generation in the green task there is always an individual in the population able to preserve some knowledge about the red task. The retention never drops below 10. The forgetting for the red task at the end of the evolution on the green task is 9.9 fitness points. We observe a similar behavior when switching back to the red task and measuring the retention on the green task (from generation 400). The forgetting in this case is 14.2 fitness point (with a final performance on the green task of around 32 points). Overall, we can see that lifelong evolution did not prevent the swarm adapting to a novel task *and* it allowed individuals in the population to preserve knowledge about prior tasks *without being explicitly evolved for this purpose.* Compared to the forgetting typically exhibited by lifelong learning agents in deep learning (Masana et al. 2023), where the network completely forgets previous task if no specific lifelong techniques are used, the lifelong evolution of swarms is much more robust to forgetting. This is likely due to the variety of the population maintained by evolutionary algorithms, like NEAT, that preserve previously discovered skills even though they may not be immediately useful (Mouret and Clune 2015; Pugh, Soros, and Kenneth O. Stanley 2016).

6.4.3. Forgetting in the individuals

Lifelong deep learning develops a single agent that needs to be capable to solve all encountered tasks. We study our lifelong evolutionary framework also from this perspective, to understand the benefits and pitfalls of an agent-centric view. We adopt the same dynamic

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

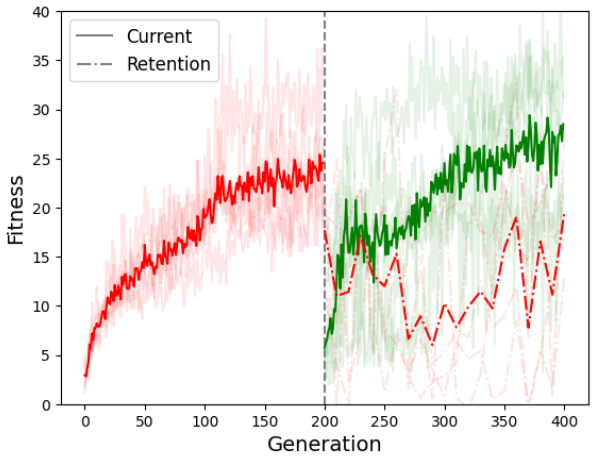


Figure 15: Current performance (solid lines) and retention at the individual level (dashed lines) for the red and green tasks when using model-specific regularization coefficients. This setup improves retention and the performance on the green task compared to Figure 14.

Table 8: Summary of results. "Red" and "Green" represent the current performance on their respective tasks. "Ret. Red" indicates the retained performance on the red task after evolving on the green task. Conversely, "Fgt. Red" measures forgetting on the red task after evolving on the green task. The approaches include population-based evolution (*pop*), top-performing individual (*ind*), regularized evolution with a fixed regularization coefficient (*reg*) and with a model-specific coefficient (*m.s. reg*).

	Red ↑	**Green** ↑	**Ret. Red** ↑	**Fgt. Red** ↓
pop	24.54	32.36	14.64	9.9
ind	24.54	32.36	2.08	22.46
reg	24.54	21.44	15.38	9.16
m.s. reg	24.54	28.5	19.52	5.02

environment of the previous sections. We select the top-performing individual on the current task and we evaluate its performance on previous tasks. Note that this approach differs from Section 6.4.2, as there we selected the individual which performed best *on the previous task*. The results, shown in Figure 13, are in stark contrast to the population-level retention. The top individual catastrophically forgets previous knowledge, with retention scores dropping to near-zero across all generations. Specifically, forgetting for the red task at generation 400 is 22.5, and for the green task at generation 600 is 19.9. Therefore, while lifelong evolution supports retention at the level of the whole population, it does not do the same for each individual. These results are aligned with the behavior commonly observed in deep lifelong learning (Masana et al. 2023).

We tweak the evolution process with our genetic regularization, choosing as the reference model the best performing individual on the red task right before the switch to the green task (generation 200). Figure 14 shows the results of applying the same regularization coefficient $\lambda = 11$ on all runs. The retention curve is notably higher compared to the one without regularization (Figure 13), reaching 15.38 at generation 400 and reducing forgetting to 9.16. In

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

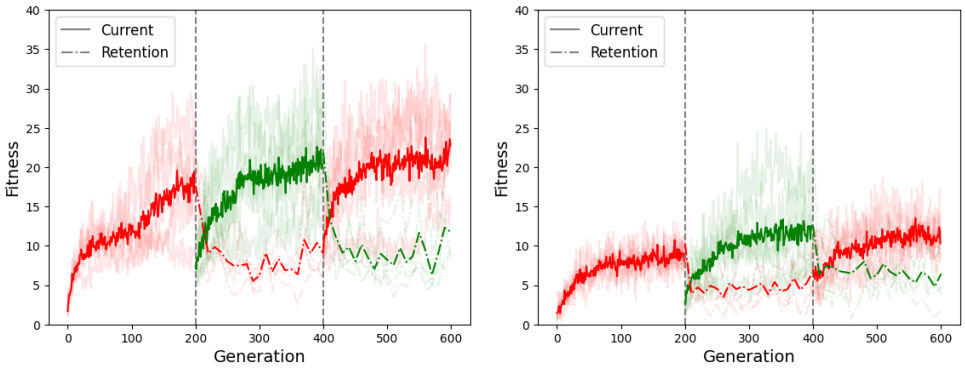


Figure 16: Impact of population scaling on current performance (solid lines) and retention at the population level (dashed lines). The top plot shows results for a population size of 100, while the bottom plot displays results for a reduced population size of 15. Although the population size significantly affects the overall fitness levels (higher fitness with a larger population), a smaller population is still able to preserve some knowledge about previous tasks.

contrast, the prior experiment exhibited a complete performance drop with a forgetting of 22.5.

Aligned with the deep lifelong learning literature (Parisi et al. 2019; Kirkpatrick et al. 2017), the regularization comes with a trade-off: the performance on the green task decreased from 32.36 (without regularization) to 21.44.

Model-specific regularization So far, we used the same λ coefficient for all runs. However, the coefficient directly impacts on architectural changes and, as such, different architectures may need different values of λ. To test this hypothesis, we tuned λ on each run separately (each run using a different random seed). Aligned with our expectations, Figure 15 shows that we can indeed improve both retention and the performance on the current task with a model-specific λ. The retention for the red task at generation 400 reaches 19.52, reducing forgetting to just 5.02.

We select the top-performing controller from the model-specific regularization runs and evaluate it across 100 environments per task. It achieves a fitness score of 28.5 on the green task and 26.1 on the red task, with a forgetting score of only 2.2, demonstrating remarkable multitask learning capabilities (Caruana 1997).

Table 8 summarizes our results. The model-specific regularization retained the most knowledge at the expenses of a slight performance drop on the green task. Even with a fixed λ value, a favorable balance between retention and current performance was achieved. At the population level, the controller is able to adapt to new tasks and to preserve some knowledge about previous ones.

6.4.4. Stress tests on the population

We turn our attention to two key factors of our lifelong evolutionary process: the population size and the number of task drifts.

Reducing the population size. We study whether smaller populations are still able to preserve previous knowledge. Starting from the same dynamic environment of Section 6.4.2, we

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

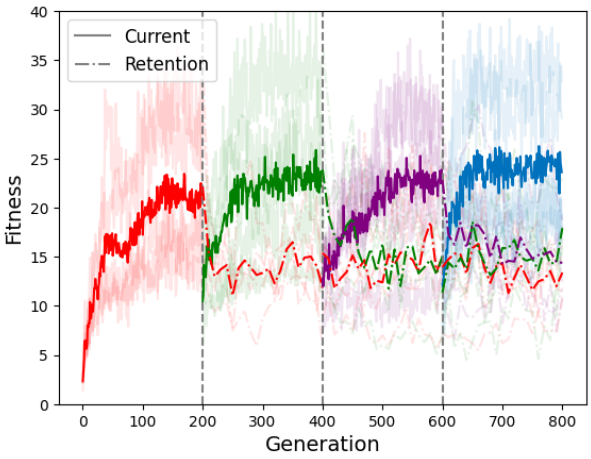


Figure 17: Current performance (solid lines) and retention at the population level (dashed lines) across four sequential tasks (red, green, purple, and blue). The population maintains sufficient diversity to retain knowledge on all previous tasks, while adapting to new ones.

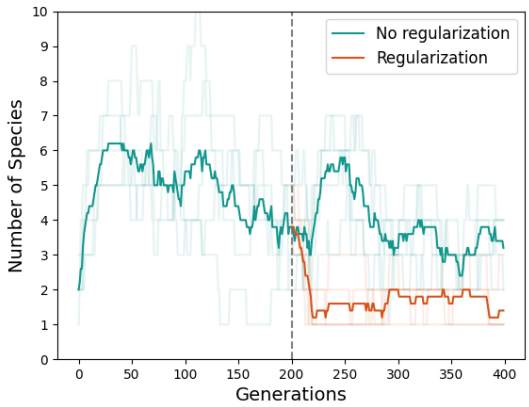


Figure 18: Number of species over generations with and without regularized evolution.

reduce the population from 300 to 100 individuals (Figure 16, top) and as low as 15 individuals (Figure 16, bottom). For the former case, forgetting is 9.38, similar to what we achieved for a population of 300. For the latter, forgetting is 1.38 but only because the performance on the first task is very low. Therefore, smaller populations forget less only because they are unable to adapt properly. When the population size enables effective adaptation, forgetting remains comparable.

Increasing the number of drifts. Using a population of 300 individuals, we increase the number of drifts to include four different tasks: red, green, purple and cyan. Since we need to distinguish between more colors (eight in total, two per task), the input neurons for the neural networks are increased correspondigly (Table 7).

We evaluate the performance on *all* previous tasks at the population level. Figure 17 shows that the population successfully retains previous knowledge at a similar score, without being significantly impacted by the addition of new tasks.

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Variation of species. We analyze the number of NEAT species present in the population at each generation to understand how its diversity changes over time (Figure 18). When evolving without regularization the evolution starts with around two species, which increase to 6 by generation 50. Over time, species with stagnating performance are gradually discarded. This behavior repeats when the task changes. In contrast, the regularized evolution shows a significant reduction in species diversity after the task change, when the genetic distance penalization is applied. The population often converges to one or very few species, which aligns with the objective of optimizing individuals to remain similar to the single reference model. While this approach enhances retention and mitigates forgetting, it reduces diversity. Incorporating mechanisms to preserve diversity around the reference model may improve evolvability (Mengistu, Lehman, and Clune 2016; Lehman and Kenneth O. Stanley 2011) and sustain evolution over longer time spans.

We believe our empirical results provide solid foundations for future work. Other lifelong techniques may be adapted into our framework (e.g., replay (Hayes et al. 2021; Merlin et al. 2022)) to improve retention. Lifelong learning may also benefit from the collective approach of evolutionary algorithms, with opportunities to naturally mitigate forgetting by maintaining an archive of previously useful skills (Lehman and Kenneth O. Stanley 2011) in the form of compressed models. Lifelong evolutionary approaches can have an impact well beyond lifelong learning and swarm intelligence alone. Letting the evolution and learning processes co-design an adaptive system is indeed one of the most promising and interesting research directions we can envision for the future.

7. Preliminary results on training modular ensembles of RNNs

Modularity has emerged as a crucial mechanism for building complex, integrated systems from simpler, “primitive” learning modules. By leveraging the representational capacity of these individual submodules, compositional learning enables the formation of richer and more sophisticated representations. This concept mirrors the current understanding of the human brain where, while sub-areas are delegated to specific tasks, the interactions among the areas provide enhanced and more complex representations Lake et al. 2017. Changing the perspective to one of semantic representations of concepts, combining simple concepts into more complex ones is a cornerstone of the human ability to understand, reason, and learn. Equipping learning modules with compositional abilities allows to benefit from the reuse of specialized knowledge, as well as from richer representations to solve intricate problems more accurately Sinha, Premsri, and Kordjamshidi 2024. Joining the notion of compositionality to the landscape of dynamical systems leads to the desiderata of achieving the ability to compose multiple dynamical systems into an assembly of dynamical systems, whose dynamics emerging from appropriate modelling of their mutual interaction enrich the information of the singles. This property is particularly beneficial in the context of sequential data processing tasks, where RNNs represent a model of choice. Interpreting these models as an input-driven dynamical system is a widely adopted practice in literature B. Chang et al. 2019 and, as such, ensuring their *stability* is a crucial aspect to address.

In the following, we outline the two main research directions we developed for modular ensembles of RNNs. In Section 7.1, we explore the construction of stable RNN assemblies through negative-feedback connections. In Section 7.2, we focus on adaptive learning of RNN modules, enabling the dynamic activation of specific modules within an RNN ensemble.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

7.1. The framework of a collective of RNNs coupled in negative-feedback

In the context of assembling intelligent sequential models together, the seminal work in Kozachkov, Ennis, and J.-J. Slotine 2022 first investigated stable assemblies of RNNs and provided solution for coupling different stable RNN modules to preserve overall stability. The main idea is to define RNNs that correspond to *contractive* dynamical systems and exploit contract theory results in the literature Lohmiller and J.-J. E. Slotine 1998 to build up larger RNNs of RNNs with stability guarantees.

We consider a number p of RNN subnetworks of the following type Kozachkov, Ennis, and J.-J. Slotine 2022:

$$\tau\dot{\mathbf{x}}_i = -\mathbf{x}_i + \mathbf{W}_i\phi(\mathbf{x}_i) + \mathbf{u}_i(t), \qquad i = 1, \dots, p, \tag{46}$$

where $\mathbf{x}_i$ is the hidden state of the i-th RNN, $\mathbf{W}_i \in \mathbb{R}^{N\times N}$ are the recurrent connections, $\mathbf{u}_i(t)$ the input driving the i-th subnetwork, and ϕ is the nonlinear activation function, $\tanh$ in our experiments.

Hierarchical coupling The simplest way to couple a collective of stable RNNs into a stable RNN of RNNs is to introduce feedforward connections from one RNN to another, forming a deep hierarchical RNN architecture. If the single RNN modules are stable, then by Lemma 2 and Theorem 1 of Gallicchio and Micheli 2017, we get that the deep hierarchical RNN is stable.

Enabling feedback connections can make the overall system dynamics potentially unstable. However, this challenge can be addressed through architectural techniques, such as negative-feedback couplings.

Negative-feedback coupling We couple the i-th RNN module with the j-th RNN module them via matrices $\mathbf{L}_{ij} \in \mathbb{R}^{N\times N}$ with the skew-symmetric constraint as follows:

$$\mathbf{L}_{ij} = -\mathbf{L}_{ji}^T, \tag{47}$$

so that the overall RNN of RNNs defined by:

$$\tau\dot{\mathbf{x}}_i = -\mathbf{x}_i + \mathbf{W}_i\phi(\mathbf{x}_i) + \sum_{j=1}^{p}\mathbf{L}_{ij}\mathbf{x}_j + \mathbf{u}_i(t), \qquad i = 1, \dots, p, \tag{48}$$

is guaranteed to be a stable system whenever the single RNN modules in (46) are themselves stable Kozachkov, Ennis, and J.-J. Slotine 2022.

Their theoretical investigation is accompanied by the proposal of approaches to guarantee the contractive dynamics of the single RNN modules accordingly. A first strategy relies on complex optimization schemes to allow the adaptation of the RNN modules and the connections between them with stability guarantees. However, a second, more straightforward and better-performing strategy, denoted as Sparse Combo Net, consists of simply *fixing the RNN modules' recurrent weights in a contractive configuration* and only learning the connections between RNN modules. These results highlight how it remains an open and pressing problem to find strategies for adapting the internal weights of the RNN modules to outperform the simple strategy of keeping them fixed.

Sparse Combo Net SCN initializes the weights $\mathbf{W}_i$ in eq. (46) according to the criterion in Kozachkov, Ennis, and J.-J. Slotine 2022, Theorem 1 to achieve a contractive

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

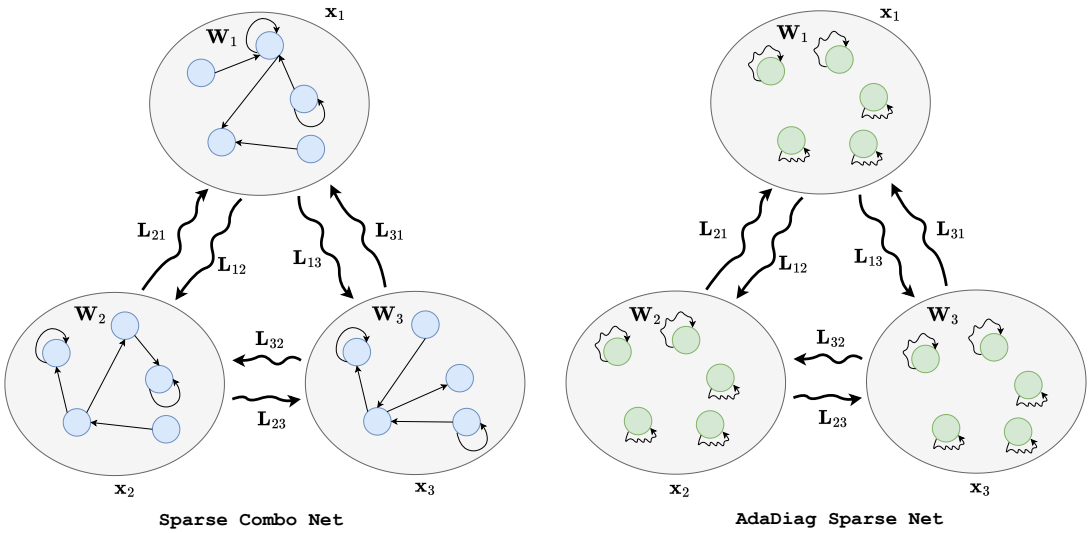


Figure 19: Depiction of an assembly of RNNs. Straight arrows denote untrained connections, while wavy arrows denote trained connections. **Left**: Sparse Combo Net trains only the connections between RNN modules leaving the internal connections of the single RNN modules untrained. **Right**: AdaDiag Sparse Net (our) trains the connections between RNN modules and also the internal connections of the single RNN modules, which result in adaptive self-loops.

parameterization. Factually, such a criterion is too computationally intensive to be checked at each weights-update iteration, they are left untrained throughout the learning process, while the coupling matrices $\mathbf{L}_{ij}$ are trained under the constraint of eq. (47).

7.1.1. Advancements on trainable RNNs of RNNs

Contrary to SCN, we take a step further and focus on constructing *stable* and *fully adaptive* assemblies of RNNs, while relaxing from the complexity arising from the optimization of the RNN modules. We propose a couple of strategies for adapting the internal weights of RNN modules while ensuring contraction dynamics during training. We use the same methodology as SCN for training the coupling matrices $\mathbf{L}_{ij}$, but we allow the internal connections of the RNN modules to be trained. Both these strategies ensure that the spectral norm of the matrix $\mathbf{W}_i$ is less or equal than 1 during training. This guarantees that each single RNN module, if decoupled from all the others, is contracting Yildiz, Jaeger, and Kiebel 2012. Both our proposals involve the use of single RNN modules whose internal units do not communicate with each other, i.e. the matrices $\mathbf{W}_i$ are structured to be diagonal. We call this model *AdaDiag Sparse Net*. Restricting individual RNN modules to a diagonal form limits their expressiveness, as their neurons remain independent. However, the overall network preserves its representational ability since the coupling matrices $\mathbf{L}_{ij}$ enable communication between neurons across different modules.

AdaDiag Sparse Net with tanh. We consider diagonal matrices $\mathbf{W}_i$ with entries constrained via the component-wise application of the hyperbolic tangent on the entries of $\mathbf{W}_i$. This strategy ensures that the diagonal elements of $\mathbf{W}_i$ assume values in $(-1, 1)$. Therefore, the spectral norm of $\mathbf{W}_i$ is necessarily less than 1.

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Block	Method	sMNIST		psMNIST	
		$C = 5$	$C = 20$	$C = 5$	$C = 20$
SCN	Sparse (3%)	$77.99_{\pm 1.58}$	$\mathbf{90.04_{\pm 0.72}}$	$74.32_{\pm 2.51}$	$82.08_{\pm 0.98}$
SCN	Sparse (30%)	$85.44_{\pm 1.67}$	$90.04_{\pm 0.92}$	$80.18_{\pm 0.51}$	$85.91_{\pm 1.36}$
AdaDiag (our)	$\tanh$	$86.60_{\pm 0.68}$	$88.71_{\pm 0.07}$	$\mathbf{85.11_{\pm 0.25}}$	$88.53_{\pm 1.23}$
AdaDiag (our)	clip	$\mathbf{87.74_{\pm 0.11}}$	$88.89_{\pm 0.06}$	$84.27_{\pm 1.10}$	$\mathbf{89.63_{\pm 0.25}}$
Vanilla RNN	dense	49.10	–	–	71.60

Table 9: Performance of different configurations of assemblies (including number of coupling blocks C) on sMNIST and psMNIST. The first two rows correspond to two sparsity settings of the SCN model found in Kozachkov, Ennis, and J.-J. Slotine 2022. The third and fourth rows correspond to our methods. In the last row, the performance of a fully-connected RNN trained as a monolithic block, with a comparable number of trainable parameters. We report the mean and standard deviation of the test set accuracy averaged over three runs, apart from Vanilla RNN which is taken from S. Chang et al. 2017. The best result for each dataset and coupling block configuration is highlighted in bold.

AdaDiag Sparse Net with clip. We consider diagonal matrices $\mathbf{W}_i$ with entries clipped to 0.99 whenever they exceed the value 1, or clipped to -0.99 whenever they assume a value less than -1. This strategy ensures that the diagonal elements of $\mathbf{W}_i$ assume values in $(-1, 1)$. Therefore, the spectral norm of $\mathbf{W}_i$ is necessarily less than 1.

Experimental results The purpose of our experiments is to provide an analysis of the proposed strategies in comparison with the best model from Kozachkov, Ennis, and J.-J. Slotine 2022 as baseline. In our setup, the assembly size is fixed to 16 recurrent modules, each consisting of 32 units. For an assembly of 16 modules, the total amount of possible coupling blocks is 240, but under the constraint of eq. (47), the total trainable coupling blocks is 120 (i.e., $\frac{16\times 15}{2}$). We analyzed the behaviour under different levels of sparsity in the modules' coupling by setting the number of coupling blocks C to 5 and 20. To pursue the aforementioned objective, we configured the initialization and the adaptivity of diagonal blocks in the following four ways: (1) fixed, sparse matrix with 3% of nonzero entries (as in Kozachkov, Ennis, and J.-J. Slotine 2022); (2) fixed, sparse matrix with 30% of nonzero entries; (3) diagonal matrix adapted with strategy 1; (4) diagonal matrix adapted with strategy 2. The nonlinearity in eq. (48) is $\tanh$, and the discretization step is 0.03 (using forward Euler method). We assessed all the configurations on the regular and the permuted version of Sequential MNIST. We trained and validated each configuration on the given train/test split three times. Each run was limited to a maximum of 200 training epochs, and we applied early stopping when reaching a plateau in the validation accuracy. In Table 9, we report the performance of all the assessed configurations, plus the performance of a Vanilla RNN as reference. Starting from the configurations with lower coupling among the RNN modules, i.e., with $C = 5$, we can observe that our model outperforms the baseline by $\sim 2\%$ with the clipping method on sMNIST and by $\sim 5\%$ with the $\tanh$ method on psMNIST. From these results, we deduce that the accuracy benefits from the adaptivity arising from the interaction between the models when the coupling is low. Even more relevant, when the coupling is higher, i.e., with $C = 20$, we experience a significant improvement in the performance on psMNIST, as

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

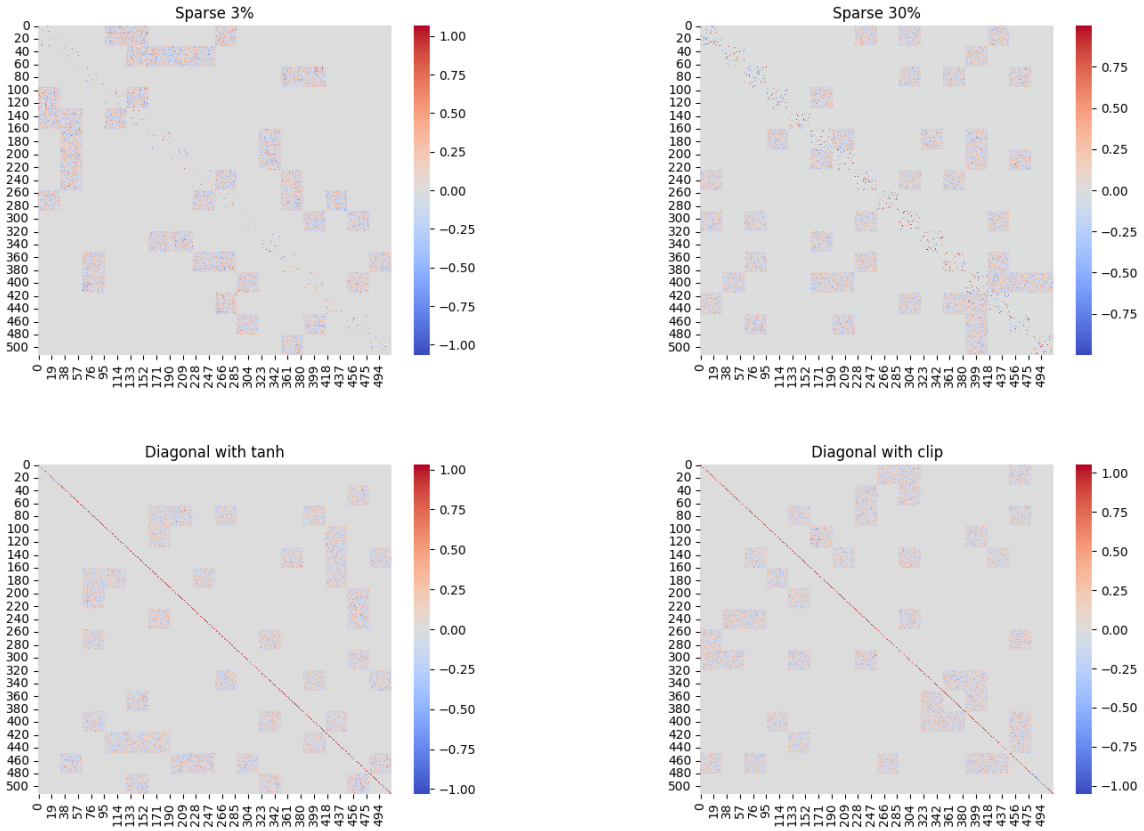


Figure 20: Representation of the weights of the RNN of RNNs for each strategy on psMNIST with $C = 20$.

the adaptivity of the RNN modules tackles the higher complexity of the task. From a general perspective, we observe that our strategies exhibit greater consistency in performance across runs, as evidenced by a significantly lower standard deviation compared to Sparse Combo Net in most cases. This improved stability can be attributed to the adaptivity of the RNN modules, which help mitigate the impact of poor parameterization during the initialization phase. This is further supported by the representation of the weights in Figure 20, where we can observe that the entries of the coupling blocks tend to saturate more on the Sparse Combo Net. Hence, we conjecture that adapting the inner dynamics of each RNN modules by accounting also for the interactions with other networks allows to better accomodate the knowledge across the assembly, ultimately leading to a performance improvement. We remark that adapting the RNN modules in the training phase does not produce a significant overhead from a computational perspective, as the Sparse Combo Nets required an average time per epoch of $\sim 185s$, against the $\sim 206s$ of our strategies. Finally, we conclude by noticing that, training as a monolithic block a Vanilla RNN results in poorer performance in the considered classification benchmarks, despite its theoretically greater expressive power due to lack of architectural constraints.

7.2. Integrating attention into reservoir-based assemblies

Our objective for this task is to combine several intelligent models, to enhance their dynamical adaptability to different inputs. After performing different experiments on the composition of multiple models, we propose a compositional approach based on Echo State Networks.

The major inspiration was the work from Goyal et al. 2019, in which the authors propose models

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

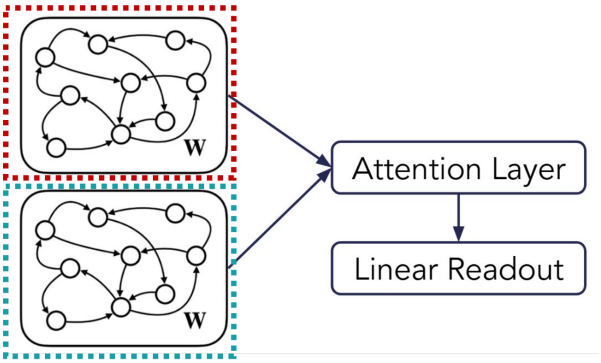


Figure 21: ESNs Composition. In this case, a linear model (in red) and a non-linear one (in blue) are composed via an attention layer. A linear readout is used to perform the classification. We experimented with both a neural linear classifier and a ridge regression model.

composition through attention layers. In this paper, they present the *Recurrent Independent Mechanisms* (RIM), a modular architecture designed for out-of-distribution generalization with recurrent models, like LSTM and GRU.

The RIM framework is built around three fundamental phases. In the first one, an attention mechanism is employed to select the most relevant modules for each input sequence. After that, each recurrent module runs its independent dynamics, given its hidden state. Lastly, the *k* selected modules are updated through a second attention mechanism, that acts as a communication layer across RIMs. In this way, each module can attend useful information encoded in other modules' hidden states.

In the paper, the authors show that such modular approach is able to better generalize to out-of-distribution samples at inference time, with experiments on a variety of tasks.

Building on this idea, we propose an attention-based ESN composition model. Our intuition is that, when dealing with dynamic scenarios, a single model could not be enough to solve every task. In particular, scenarios where samples from different tasks arrive interleaved over time are common in real-world situations. Such scenarios require an high adaptability, which often cannot be achieved by a single model. We propose a modular framework that enforces adaptation to different tasks via a learned composition of different modules. In these preliminary experiments, we employ two Echo State Network modules, each with different properties and dynamics. In particular, we define a "*non-linear*" ESN, *i.e.* with *tanh* activation function, and a "*linear*" one, *i.e.* with the identity activation function. In principles, the first model is more suited for tasks involving an high degree of non-linearity, while the second one performs better in tasks which require more memory capabilities.

On top of these model, an attention layer from Vaswani et al. 2017 is trained to select the right module for the task in input. We take the hidden state from each model at the last time-step, and we compute a new hidden state given by their weighted sum. The weights are computed by the attention layer. Formally, the composed hidden state $\mathbf{h}^*$ is computed as:

$$\begin{gathered}\mathbf{Q}_n = \mathbf{W}_n^q \mathbf{h}_n \qquad \mathbf{K}_n = \mathbf{W}_n^k \mathbf{h}_n \qquad \mathbf{V}_n = \mathbf{W}_n^v \mathbf{h}_n \\ \mathbf{h}^* = \text{softmax}\left(\frac{\mathbf{Q}_n (\mathbf{K}_n)^T}{\sqrt{d_k}}\right) \mathbf{V}_n\end{gathered} \tag{49}$$

where $\theta_n = (\mathbf{W}_n^q, \mathbf{W}_n^k, \mathbf{W}_n^v)$ are the attention's parameters — namely the query, key and value matrices — d_k is the key layer dimension and $\mathbf{h}_n$ is the hidden state for the module n, with N

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

	Linear ESN	Non-Linear ESN
Linear Task	0.25	0.99
Non-Linear Task	0.8	0.01

Table 10: Ablation studies showing NRMSE (↓). We show that a given model, when applied to its corresponding task, has the best results. However, a single model is not able to achieve good performance on both tasks.

indicating the total number of modules. Note that, with this formulation, a soft composition is implemented, rather than the hard selection of a single model. Figure 21 shows an overview of the proposed model.

7.2.1. Experiments

To perform our experiments, we used the regression task from Inubushi and Yoshimura 2017, where a strong *memory-nonlinearity* trade-off is introduced:

$$y_t = \sin(\nu x_{t-\tau}) \tag{50}$$

We defined a mixed dataset, composed by samples from two different tasks. Here, each task is defined by setting the values of ν, which influences the degree of non-linearity, and τ, which determines the memory capabilities needed. For the *non-linear* task, we set $\tau = 1,\ \log \nu = 1.6$, while the *memory* task is created with $\tau = 20,\ \log \nu = -1.6$. For ease of implementation, but without loss of generality, we used a *windowed* approach: we consider each sequence in the dataset as a portion of the entire sequence y, and for each portion we compute the target value on the last timestep. We created 50000 sequences of length 1000 for each task, using 20% of the dataset as a test set and the remaining sequences as training set.

First, we conducted a model selection process for both tasks to determine the optimal configuration for the *memory* and *non-linear* models. We fixed the hidden size of the models at 100 and focused primarily on the input scaling and spectral radius hyperparameters, as they had the most significant impact. For each task, these values were randomly sampled from a uniform distribution, with input scaling constrained to $[0.2, 6]$ and the spectral radius to $[0.1, 3]$. As a result of this hyperparameter search, we obtained:

- *Memory* task $\rightarrow$ input scaling $= 1.3$; spectral radius $= 0.9$;
- *Non-linear* task $\rightarrow$ input scaling = $= 3.2$; spectral radius $= 0.7$.

Using these hyperparameters, we trained our attention-based model. Preliminary experiments revealed that the best performance was achieved with an attention key and query size of 128 and a learning rate of 10^{-3}. To train the attention mechanism, we utilized a dummy MLP classifier, enabling the use of the standard backpropagation algorithm during each training epoch. However, at the end of each epoch, model evaluation was conducted using a Ridge Regressor classifier. This approach resulted in a test set NRMSE of 0.032.

We also examined the attention scores on the test set to gain insights into the model's behavior. Figure 22 illustrates this behavior, where each point represents the average attention score assigned to the hidden state of the *non-linear* model across all test sequences. Since the test set was not shuffled, the first half of the samples corresponds to the *non-linear* task, while

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

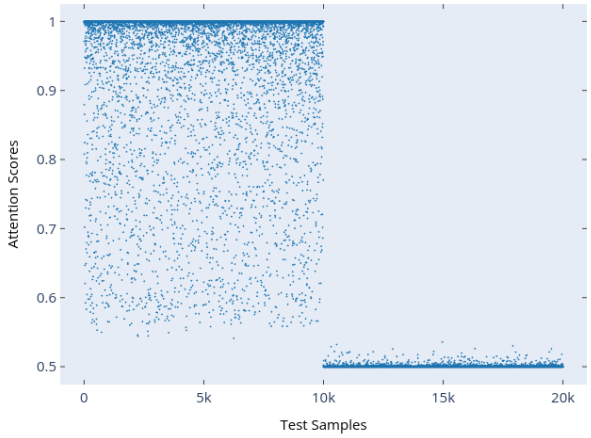


Figure 22: Attention scores given to the *non-linear* model. The first half of the test samples belong to the *non-linear* task, while the second half represents the *memory* task.

the second half represents the *memory* task. As expected, the model initially assigns high attention to the *non-linear* model, followed by a sharp shift in attention when the task changes.

In summary, these preliminary results indicate that the attention mechanism successfully selects the appropriate model based on the input task. Moving forward, we plan to further explore this approach by tackling more complex tasks and experimenting with model compositions involving a greater number of modules.

8. Preliminary results on awareness through a modular composition of RNNs

In this section, we explore the emergence of temporal and spatial awareness within collectives of recurrent neural networks. This study represents an initial step toward a broader investigation into the development of awareness, with a particular focus on the temporal and spatial dimensions. These aspects are especially relevant, as they align naturally with baseline models like RNNs, which are inherently designed to process sequential spatio-temporal data. In future work, we aim to extend this analysis by incorporating archetype networks, enabling a more comprehensive understanding of the mechanisms underlying emergent awareness in the context of the ACDS framework.

8.1. Temporal awareness in RNNs

We bind the notion of temporal awareness with two main tasks of recurrent models: *time series forecasting* and *Memory Capacity* (MC). The former is the task of predicting future values of a sequential signal based on its past observations. The challenge in time series forecasting lies in capturing temporal dependencies, where past values influence future outcomes, thus requiring the acknowledgement of contextual information through time. The latter measures the ability of recurrent models to store and retrieve past inputs, measuring how well it is able to reconstruct past inputs from its current state, typically evaluated using a linear regression task

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

on delayed input signals. The MC of a recurrent model is given by:

$$MC = \sum_{d=1}^{\infty} C_d \tag{51}$$

where C_d represents the squared correlation coefficient for delay d, defined as:

$$C_d = \frac{\mathrm{Var}(\hat{u}_{t-d})}{\mathrm{Var}(u_t)} \tag{52}$$

where:

- u_t is the input signal at time t,
- $\hat{u}_{t-d}$ is the RNN's reconstructed version of u_{t-d},
- $\mathrm{Var}(\cdot)$ denotes variance.

When the recurrent model is instantiated as an ESN, the theoretical upper bound of memory capacity is given by:

$$MC \leq N \tag{53}$$

where N is the number of reservoir neurons Jaeger 2002. These two aspects intertwine to define the temporal awareness of a machine learning model. Time series forecasting relies on the model's ability to interpret its current state to anticipate future values, while memory capacity determines how much past information the model can retain and utilize. Together, they enable the model to capture temporal dependencies, ensuring informed and accurate predictions with respect to the current position in time.

When independent recurrent models, each with their own local temporal awareness, communicate and exchange information, they collectively enhance overall temporal awareness. Each model retains and processes different aspects of past information, and by interacting, they can share complementary insights, reducing individual limitations in memory and forecasting ability. This collaborative dynamic allows the system to integrate multiple perspectives on past and future dependencies, leading to a more comprehensive and robust understanding of temporal patterns.

8.1.1. Experiments

To demonstrate this property, we designed an experimental setup involving two primary ESNs, with 100 units each and spectral radius of the recurrent transformation 0.9. The experiment consists of three phases, each progressively increasing the level of interaction between the models to assess their impact on temporal awareness.

Phases description In the first phase, both *ESNs operate independently*, each trained in a standalone manner using ridge regression. This allows us to evaluate their individual memory capacity and forecasting capabilities without any shared information.

In the second phase, we integrate the representations by placing the *two reservoirs in parallel* and training a common readout layer. This setup enables the readout to leverage complementary information from both reservoirs, leading to an improvement in overall temporal awareness. The enhanced forecasting accuracy and increased effective memory capacity indicate that integrating distinct representations enriches the system's ability to capture temporal dependencies.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Model	Random		Mackey-Glass	
	MC	Forecast	MC	Forecast
ESN_1	28.52	0.227	146.19	0.0019
ESN_2	25.31	**0.220**	141.95	0.0006
$Parallel$	33.11	0.229	283.39	0.0013
$Merge$	**43.08**	0.221	**291.06**	**0.0001**

Table 11: Results of the benchmarks. The acronym MC indicates the memory capacity and the corresponding correlation metric is reported. The forecast metric is the MSE. Best results are reported in bold.

In the third phase, we *introduce direct communication between the two reservoirs* by applying a linear transformation to their respective states. This modification allows information to flow between the ESNs, creating an emergent temporal awareness that surpasses the capabilities observed in the previous phases. The improvement in both forecasting accuracy and memory capacity confirms that enabling recurrent models to exchange information enhances their ability to store past inputs and predict future values more effectively. This result highlights that temporal awareness is not just a function of individual models but can emerge through structured interaction, leading to more expressive and robust representations of time-dependent data.

Benchmarks To assess the temporal awareness in the three phases, we employed two main benchmarks:

1. An i.i.d. random signal (i.e., the common benchmark for assessing the memory capacity);
2. The Mackey-Glass time series.

Results The results presented in Table 11 provide a clear empirical validation of how integrating and allowing communication between independent Echo State Networks (ESNs) enhances temporal awareness.

The first two rows correspond to standalone models, ESN_1 and ESN_2 , each trained separately on the two datasets: a random input sequence and the chaotic Mackey-Glass system. For both datasets, the individual ESNs exhibit moderate memory capacity (MC) and forecasting accuracy. Notably, in the random input case, ESN_1 achieves a slightly higher MC than ESN_2 , while ESN_2 achieves a marginally better forecasting performance. This suggests that different reservoirs may develop distinct local temporal awareness depending on their initialization and training, reinforcing the idea that independent recurrent models capture complementary information.

In the Parallel setup, where the two reservoirs are combined in parallel with a shared readout, we observe a significant improvement in memory capacity for both datasets (33.11 for random input and a substantial increase to 283.39 for Mackey-Glass, showing a clear exploitation of patterns of the time series). This confirms that integrating multiple representations enhances the ability to retain past inputs. Forecasting performance remains comparable to the independent models, with a slight improvement for the random dataset but a small decrease

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

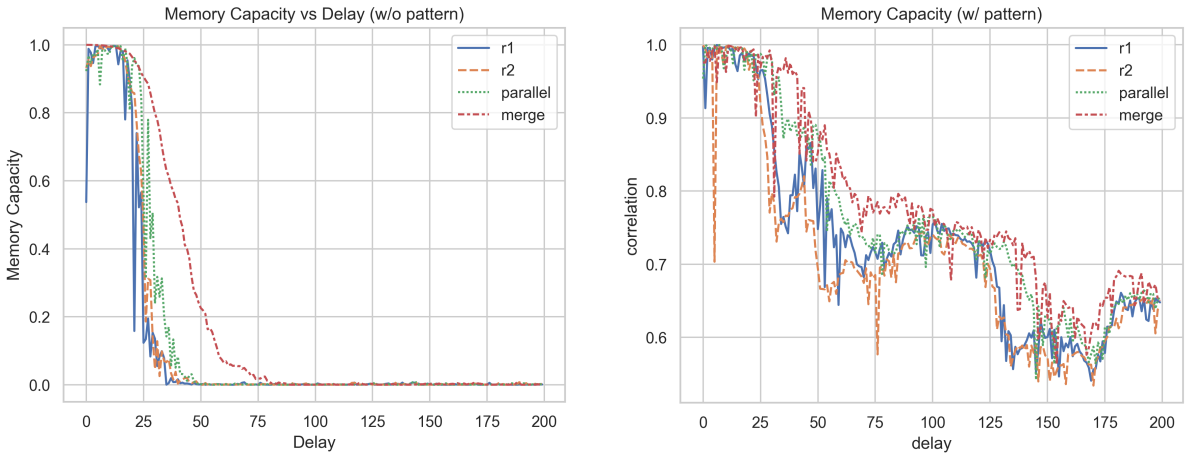


Figure 23: Plots showing the correlation coefficient in the reconstruction process with increasing delay on the Memory Capacity benchmark. The left plot shows the correlation with the random signal, while the right one shows the correlation on the Mackey-Glass time series.

for the Mackey-Glass dataset. The results suggest that while merging reservoirs provides richer representations, the readout alone may not fully exploit this additional information for prediction.

The final phase, Merge through communication, introduces communication between the two reservoirs via a linear transformation of their states. This modification yields the best overall performance across both metrics. Memory capacity reaches its peak values (43.08 for random input and 291.06 for Mackey-Glass), demonstrating that allowing information exchange between reservoirs significantly enhances the ability to store past information. This is further confirmed by plots in Figure 23, where communicating models are able to retain a significantly better reconstruction capabilities even with longer delays. Furthermore, the forecasting performance improves, with the lowest MSE (0.0001) recorded for the Mackey-Glass dataset, indicating a stronger predictive capability.

These results highlight how communicating internal representations brings further, *emergent* temporal awareness. While individual ESNs exhibit local memory and forecasting abilities, integrating their outputs improves memory capacity, and further allowing state interactions leads to an emergent global temporal awareness. This suggests that structured information exchange between recurrent models leads to richer, more expressive temporal representations, ultimately improving both the retention of past inputs and the anticipation of future values.

8.2. Spatial awareness in RNNs

Recurrent models processing time series data are inherently constrained by the spatial information available in the input sequences. Since each model relies solely on the observed signals, its internal representation and predictions remain bound to the spatial structure present in those inputs. However, time series often contain latent spatial dependencies that are not explicitly encoded in any single input stream, limiting the model's ability to capture the full structure of the system it is learning.

When multiple recurrent models with different perspectives are brought together, they can exchange information and uncover spatial patterns that neither model could deduce

European Innovation Council
Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

independently. This interaction allows them to infer missing spatial components by leveraging complementary insights from each other's learned representations. By doing so, they develop an emergent spatial awareness, which extends beyond their individual observations.

This cooperative process enhances the models' ability to recognize spatial relationships in sequential data, improving both representation learning and predictive accuracy. Rather than being constrained to the spatial information explicitly available in their respective inputs, interacting recurrent models can synthesize a more complete and structured understanding of the spatial dependencies underlying the time series.

8.2.1. Experiments

To evaluate the emergence of spatial awareness in recurrent models, we designed an experimental setup using the chaotic Lorenz attractor time series, a well-known dynamical system with three coupled differential equations that exhibit complex spatial dependencies. The goal of the experiment was to investigate how recurrent models, specifically ESNs, can infer missing spatial information when provided with partial inputs. In this setup, we test the ability of ESNs to predict the full spatial dynamics of the system by combining different perspectives of the input time series, particularly focusing on how the models can deduce spatial components that are not explicitly provided in the input.

Phases description In the first phase, we train an ESN that takes the x component of the Lorenz attractor as input and attempts to predict all three dimensions, (x, y, z). The model has to infer the missing y and z components based solely on the temporal patterns within the x series. In the second phase, a second ESN is trained on the y component of the attractor, again attempting to predict all three dimensions. This setup allows us to assess how much spatial information each individual feature carries and whether an ESN can predict the full dynamics of the system with only partial input. Finally, in the third phase, we allow the two ESNs to communicate with each other by introducing a linear transformation of their respective states. In this phase, both the x and y components are provided as inputs, and the model aims to predict all three dimensions (x, y, z). The introduction of communication between the models tests whether this interaction allows them to uncover missing spatial components and enhance the overall spatial awareness of the system.

The Lorenz system is described by the following set of ordinary differential equations:

$$\frac{dx}{dt} = \sigma(y - x)$$

$$\frac{dy}{dt} = x(\rho - z) - y$$

$$\frac{dz}{dt} = xy - \beta z$$

where x, y, z represent the three spatial dimensions of the Lorenz attractor, and the parameters σ, ρ, and β are typically set to 10, 28, and 8/3, respectively, for the chaotic system.

Results The results presented in the Table demonstrate the performance of ESNs in predicting the three dimensions (x, y, and z) of the Lorenz attractor under different input

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Model	MSE on x	MSE on y	MSE on z
x only	0.0007	0.0006	0.2212
y only	0.0020	0.0025	0.2214
x and y	**0.00016**	**0.00007**	**0.2207**

Table 12: MSE results for predicting the three dimensions of the Lorenz attractor. The Table shows the MSE for x, y, and z under different input conditions. The best results are reported in bold.

conditions. Specifically, we examined three configurations: training the model with only the x component as input, only the y component, and both x and y together. The table shows how these different setups affect the MSE for each dimension, providing insight into the model's ability to predict the missing components when only partial information is available.

When the model is trained using only the x dimension as input, it performs relatively well in predicting x and y, with low MSE values of 0.0007 and 0.0006, respectively. However, the MSE for the z-dimension remains high at 0.2212. This result highlights that although the model can capture some temporal relationships in the x and y components, the absence of the y-dimension limits its ability to predict the full spatial structure of the Lorenz attractor. Without y, the spatial dependencies between the three dimensions are not fully understood, and the model struggles to account for the missing information in z.

Similarly, training the model with only the y-dimension as input results in relatively high MSE values for both x and y (0.0020 and 0.0025), while the z-dimension's prediction remains poor (0.2214). The lack of the x-dimension restricts the model's spatial awareness in much the same way as the x-only case. This suggests that a single dimension is insufficient for capturing the full spatial dependencies in the system, and the model needs at least two complementary inputs to build a more accurate spatial representation.

When both x and y are provided as inputs, the performance on the z-dimension remains similar, but the interesting behavior is observed in the predictions for x and y. For these two variables, the MSE shows a drastic reduction compared to the models that take only x or only y as inputs. This significant improvement in the MSE of x and y, qualitatively reflected also in Figure 24, highlights the model's enhanced ability to capture hidden information embedded in the spatial relationship between the two dimensions. It demonstrates how combining multiple perspectives allows the model to better understand the complex interdependencies between x and y, leading to a much more accurate prediction for both dimensions.

These results highlight the emergence of spatial awareness in recurrent models. When provided with only a single spatial dimension, the model lacks the ability to fully capture spatial dependencies. However, when both x and y are available, the model exhibits a significantly improved understanding of their relationships, leading to a drastic reduction in prediction errors for both dimensions. This suggests that spatial awareness *emerges* as the model integrates complementary inputs, leveraging hidden dependencies that would otherwise remain inaccessible. Ultimately, enabling models to interact with different spatial perspectives enhances their ability to uncover and exploit underlying spatial structures.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

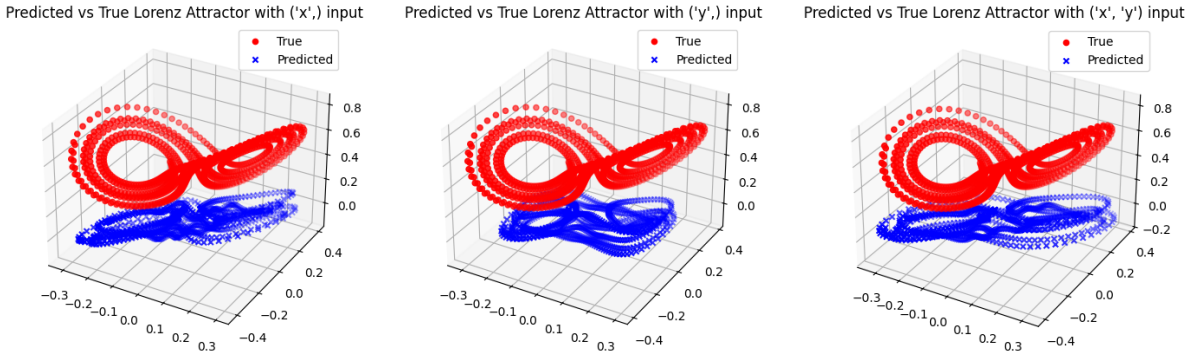


Figure 24: Representation of the predictions of the Lorenz system under different input conditions.

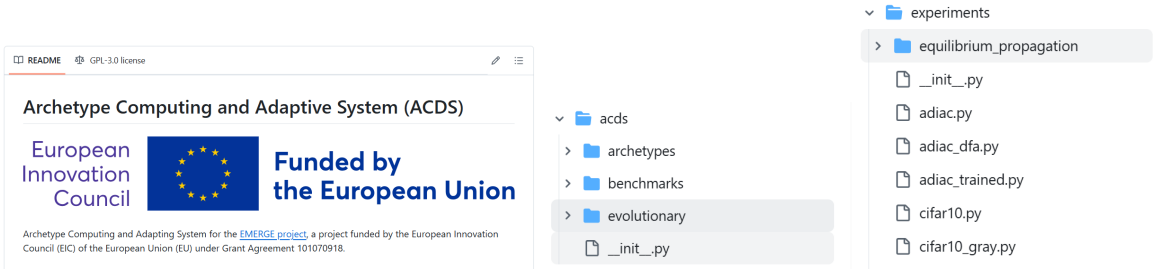


Figure 25: Archetype Computing and Adaptive System (ACDS) library, publicly available on GitHub (left). The “archetypes”, “benchmarks” and “evolutionary” modules (middle) together with the “equilibrium propagation” and experiment files on trained and untrained archetype networks (right).

9. Software library

The Archetype Computing and Adaptive System (ACDS) library is available via GitHub at this URL (Fig. 25): `https://github.com/EU-EMERGE/archetype-computing-adaptive-system`.

With respect to the status reported in D4.1 (to which the interested reader is referred to for a general description of the repository), the ACDS library has received several improvements from new contributors. Overall, the library features 6 contributors with 7 Pull Requests:

- New additions to the **archetypes** include the Physically-Implementable RON archetype network and both the Antisymmetric RON and Antisymmetric Physically-Implementable RON networks, discussed within this document in Section 2 and 2.4, respectively. All the archetype networks can be found within the “acds” module of the library, in the “archetypes” folder. The implementation of RON and an example usage is shown in Figure 26 and 27, respectively.
- The library also implements the **Lifelong Evolutionary framework for swarms** (within the module “acds/evolutionary”) proposed and discussed in Section 6.
- It is now possible to train recurrent neural networks with the **DFA** algorithm discussed in Section 4.2 (Figure 28). The corresponding experiment can be found in the “experiments” module.
- Similarly, the module “equilibrium propagation” collects our experiments using EP to train archetype networks.
- The library also includes **new benchmarks** within the “acds/benchmarks” module:

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

RON archetype network

```python
def cell(
    self, x: torch.Tensor, hy: torch.Tensor, hz: torch.Tensor
) -> Tuple[torch.Tensor, torch.Tensor]:
    """Compute the next hidden state and its derivative.

    Args:
        x (torch.Tensor): Input tensor.
        hy (torch.Tensor): Current hidden state.
        hz (torch.Tensor): Current hidden state derivative.
    """
    hz = hz + self.dt * (
        torch.tanh(
            torch.matmul(x, self.x2h) + torch.matmul(hy, self.h2h) + self.bias
        )
        - self.gamma * hy
        - self.epsilon * hz
    )

    hy = hy + self.dt * hz
    return hy, hz
```

Figure 26: RON archetype network implementation in the ACDS library.

Application of RON on an MNIST time series

```python
for images, labels in tqdm(train_loader):
    images = images.to(device)
    images = images.view(images.shape[0], -1).unsqueeze(-1)
    output = model(images)[-1][0]
    activations.append(output.cpu())
```

Figure 27: RON archetype network applied on an input time series from MNIST with the ACDS library.

Libras, CIFAR10 (both RGB and grayscale versions), Mallat and Trace.

10. Conclusions

This deliverable completes the development of the Archetype Computing System (ACS) and introduces the first implementation of the Archetype Adapting System (ADS), finalizing the core computational framework of WP4 in the EMERGE project. The ACS has been extended with new classes of archetype networks that improve its theoretical properties and practical relevance. In particular, we introduced antisymmetric Random Oscillator Networks (aRON) and their physically implementable variants (PI-RON, aPI-RON), which are theoretically grounded and empirically validated to enhance stability, expressivity, and physical deployability. We also proposed hybrid architectures such as S-RON, integrating spiking neural dynamics with oscillator-based computation. The results showcase the ability of dynamical systems built from archetype networks to *efficiently* process time-varying inputs in an effective manner, solving complex classification tasks as well as forecasting of chaotic dynamics.

On the adaptive side, we presented the first release of the ADS. We investigated biologically

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

RNN model trained with DFA

```
for epoch in range(args.epochs):
    for x, y in tqdm(train_loader):
        x = x.to(device)
        y = y.to(device).long()
        x = x.view(x.shape[0], -1, args.input_size)
        output, hidden, dW, dV, dbW, error = model(x, y=y.squeeze(-1))
        model.compute_update(hidden, dW, dV, dbW, error)
        optimizer.step()
        loss = criterion(torch.log(output), y.squeeze(-1))
```

Figure 28: An example of an RNN/GRU trained with DFA in just a few lines of code within the ACDS library.

plausible and hardware-friendly learning mechanisms, including Direct Feedback Alignment (DFA) and Equilibrium Propagation (EP), and introduced a framework for continual learning in dynamic environments through evolutionary strategies applied to swarm systems. These components enable adaptation to non-stationary tasks, with mechanisms for mitigating catastrophic forgetting and preserving diversity.

Complementary results include early investigations into modular RNN assemblies and their application to modeling temporal and spatial awareness. Together, these studies support the long-term goal of EMERGE: enabling awareness to emerge in artificial systems through the structured interaction of minimal, adaptive units.

The deliverable is supported by a unified software framework, the ACDS library, which provides the necessary tools for experimentation, development, and reproducibility of the proposed systems.

Finally, from a broader perspective, this deliverable contributes to the achievement of Milestone 4 of the project, which consolidates preliminary methods for archetype discovery, optimization, and the definition of conditions for the emergence of awareness in collectives. The work presented in D4.2 paves the way for the activities leading to Deliverable D4.3, which will complete the development of the ADS by scaling adaptation mechanisms to the network level. This progression will enable collective learning and coordination in distributed systems of archetype agents. D4.3 will be a key contribution to Milestone 6, which marks the final implementation and validation of the ACDS framework and its integration into demonstrators of collaborative awareness.

A. Proof of Proposition 1

A skew-symmetric matrix is a normal matrix, therefore, by the Spectral Theorem, there exists a $\mathbf{Q}$ unitary matrix and $\mathbf{\Lambda}$ diagonal complex-valued matrix such that $\mathbf{W} - \mathbf{W}^\top = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^*$, where * denotes the conjugate transpose. In particular, the skew-symmetric structure implies that the elements on the diagonal of $\mathbf{\Lambda}$, i.e. the eigenvalues of $\mathbf{W} - \mathbf{W}^\top$, are purely imaginary, i.e. of the form $\pm i\lambda$. All other matrices in the blocks of $\mathbf{J}_0$ are rescaled version of the identity matrix, i.e. of the form $\mu\mathbf{I}$. Hence, when summing $\mathbf{W} - \mathbf{W}^\top + \mu\mathbf{I}$ we can decompose it as

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

$\mathbf{W} - \mathbf{W}^\top + \mu\mathbf{I} = \mathbf{Q}(\mathbf{\Lambda} + \mu\mathbf{I})\mathbf{Q}^*$. Thus, we can write:

$$\mathbf{J}_0 = \begin{bmatrix} \mathbf{Q}\big(\tau^2\mathbf{\Lambda} + \big(1 - \tau^2(\delta+\gamma)\big)\mathbf{I}\big)\mathbf{Q}^* & \tau\big(1-\tau\varepsilon\big)\mathbf{I} \\ \mathbf{Q}\big(\tau\mathbf{\Lambda} - \tau(\delta+\gamma)\mathbf{I}\big)\mathbf{Q}^* & \big(1-\tau\varepsilon\big)\mathbf{I} \end{bmatrix} = \tag{54}$$

$$= \mathbf{PMP}^*, \tag{55}$$

where

$$\mathbf{M} = \begin{bmatrix} \tau^2\mathbf{\Lambda} + \big(1 - \tau^2(\delta+\gamma)\big)\mathbf{I} & \tau\big(1-\tau\varepsilon\big)\mathbf{I} \\ \tau\mathbf{\Lambda} - \tau(\delta+\gamma)\mathbf{I} & \big(1-\tau\varepsilon\big)\mathbf{I} \end{bmatrix}, \tag{56}$$

and $\mathbf{P}, \mathbf{P}^*$ are the unitary block matrices:

$$\mathbf{P} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{bmatrix}, \tag{57}$$

$$\mathbf{P}^* = \begin{bmatrix} \mathbf{Q}^* & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}^* \end{bmatrix}. \tag{58}$$

(59)

$$\mathbf{M} = \begin{bmatrix} \tau^2\mathbf{\Lambda} + \big(1 - \tau^2(\delta+\gamma)\big)\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \big(1-\tau\varepsilon\big)\mathbf{I} \end{bmatrix} + \tag{60}$$

$$+ \begin{bmatrix} \mathbf{0} & \tau\big(1-\tau\varepsilon\big)\mathbf{I} \\ \tau\mathbf{\Lambda} - \tau(\delta+\gamma)\mathbf{I} & \mathbf{0} \end{bmatrix} = \tag{61}$$

$$= \mathbf{L} + \mathbf{E}. \tag{62}$$

Given the decomposition: $\mathbf{J}_0 = \underbrace{\mathbf{PLP}^*}_{\text{diagonalizable}} + \underbrace{\mathbf{PEP}^*}_{\text{perturbation}}$, Bauer-Fike theorem Bauer and Fike 1960 implies that if μ is an eigenvalue of $\mathbf{J}_0$ then it stands at distance at most $||\mathbf{PEP}^*||$ from the diagonal values of the diagonal matrix $\mathbf{L}$, i.e. from the set of values $\{1-\tau\varepsilon\} \cup \{1-\tau^2(\delta+\gamma \pm i\lambda) : i\lambda \text{ is an eigenvalue of } \mathbf{W} - \mathbf{W}^\top\}$. Now, it remains to estimate the norm $||\mathbf{PEP}^*||$. First we notice that, since the matrix $\mathbf{P}$ is unitary then $||\mathbf{PEP}^*|| = ||\mathbf{E}||$. Finally, notice that

$$\mathbf{E} = \begin{bmatrix} \mathbf{0} & \tau\big(1-\tau\varepsilon\big)\mathbf{I} \\ \tau\mathbf{\Lambda} - \tau(\delta+\gamma)\mathbf{I} & \mathbf{0} \end{bmatrix} = \tag{63}$$

$$= \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \tau\big(1-\tau\varepsilon\big)\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \tau\mathbf{\Lambda} - \tau(\delta+\gamma)\mathbf{I} \end{bmatrix}. \tag{64}$$

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Therefore, $||\mathbf{E}|| \leq || \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} || \max\{||\tau\big(1-\tau\varepsilon\big)\mathbf{I}||, ||\tau\mathbf{\Lambda}-\tau(\delta+\gamma)\mathbf{I}||\} = \tau \max\{|1-\tau\varepsilon|, |\pm i\lambda - (\delta+\gamma)|\} = \tau \max\{|1-\tau\varepsilon|, \sqrt{\lambda_{\max}^2+(\delta+\gamma)^2}\}$.

B. Proof of Theorem 1

Necessary condition for aRON stability is that the set of values $\{1-\tau\varepsilon\} \cup \{1-\tau^2(\delta+\gamma \pm i\lambda) : i\lambda \text{ is an eigenvalue of } \mathbf{W}-\mathbf{W}^\top\}$, derived in Proposition 1, is entirely contained inside the unit circle. The farthest of these points from the origin is either $1-\tau\varepsilon$ or $1-\tau^2(\delta+\gamma+i\lambda_{\max})$, where $\lambda_{\max}$ is the spectral radius of the skew-symmetric matrix $\mathbf{W}-\mathbf{W}^\top$. It follows that a necessary condition for aRON stability is:

$$\max\{|1-\tau\varepsilon|, \sqrt{\big(1-\tau^2(\delta+\gamma)\big)^2+\lambda_{\max}^2}\} \leq 1. \tag{65}$$

In particular, it must hold that $|1-\tau\varepsilon| \leq 1$, and $\sqrt{\big(1-\tau^2(\delta+\gamma)\big)^2+\lambda_{\max}^2} \leq 1$. The first implies that $\varepsilon \leq \frac{2}{\tau}$, while the second implies that $\big(1-\tau^2(\delta+\gamma)\big)^2+\lambda_{\max}^2 \leq 1$, i.e. that $\lambda_{\max}^2 \leq \tau^2(\delta+\gamma)\big[2-\tau^2(\delta+\gamma)\big]$, from which it follows that $2-\tau^2(\delta+\gamma) \geq 0$. Moreover, each eigenvalue $i\lambda$ of $\mathbf{W}-\mathbf{W}^\top$ satisfies $|\lambda| \leq |\lambda_{\max}| \leq \tau\sqrt{(\delta+\gamma)\big[2-\tau^2(\delta+\gamma)\big]}$.

C. Proof of Proposition 2

We start by proving that the spectral radius $\lambda_{\max}$ of $\mathbf{W}-\mathbf{W}^\top$ is upper-bounded by $2||\mathbf{W}||$. Notice that, for any normal matrix $\mathbf{M}$, it holds that $\rho(\mathbf{M}) = ||\mathbf{M}||$. Now, $\mathbf{W}-\mathbf{W}^\top$ is a skew-symmetric matrix, and in particular, a normal matrix. Thus, $\lambda_{\max} = \rho(\mathbf{W}-\mathbf{W}^\top) = ||\mathbf{W}-\mathbf{W}^\top|| \leq ||\mathbf{W}|| + ||\mathbf{W}^\top|| = 2||\mathbf{W}||$. From Proposition 1, we know that a stable aRON model must have recurrent weights $\mathbf{W}$ such that the spectral radius, $\lambda_{\max}$, of $\mathbf{W}-\mathbf{W}^\top$ is upper-bounded by $\tau\sqrt{(\delta+\gamma)\big[2-\tau^2(\delta+\gamma)\big]}$. Therefore, since $\lambda_{\max} \leq 2||\mathbf{W}||$, if $2||\mathbf{W}|| \leq \tau\sqrt{(\delta+\gamma)\big[2-\tau^2(\delta+\gamma)\big]}$, then it is ensured the necessary condition of stability of aRON. We want to estimate what is the maximum possible value for $\tau\sqrt{(\delta+\gamma)\big[2-\tau^2(\delta+\gamma)\big]}$, so that we can impose a threshold on the norm of $||\mathbf{W}||$ representing a necessary condition for the stability of the aRON model. Now, $\tau\sqrt{(\delta+\gamma)\big[2-\tau^2(\delta+\gamma)\big]} = \sqrt{\alpha(2-\alpha)}$, where $\alpha = \tau^2(\delta+\gamma)$. Thus, the maximum is reached at $\alpha = 1$, and it gets the value of $\sqrt{\alpha(2-\alpha)} = 1$. Therefore, we get the necessary condition $||\mathbf{W}|| \leq \frac{1}{2}$.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

D. Details on experimental settings

E. Derivation of DFA for Gated Recurrent Unit network

We provide the update rule of DFA for all the parameters of the GRU.

$$W_z \leftarrow W_z - \eta \sum_{t=1}^{T} (Be \odot h_{t-1} - Be \odot c_t) \odot (r_t \odot (1 - r_t)) h_{t-1}^T,$$

$$V_z \leftarrow V_z - \eta \sum_{t=1}^{T} (Be \odot h_{t-1} - Be \odot c_t) \odot (r_t \odot (1 - r_t)) x_t^T,$$

$$W_r \leftarrow W_r - \eta \sum_{t=1}^{T} (W_r(Be \odot (1 - z_t)) * (1 - c_t \odot c_t) h_{t-1}) \odot (r_t \odot (1 - r_t)) h_{t-1}^T,$$

$$V_r \leftarrow V_r - \eta \sum_{t=1}^{T} (W_r(Be \odot (1 - z_t)) * (1 - c_t \odot c_t) h_{t-1}) \odot (r_t \odot (1 - r_t)) x_t^T,$$

$$W_c \leftarrow W_c - \eta \sum_{t=1}^{T} (W_r(Be \odot (1 - z_t)) * (1 - c_t \odot c_t) (r_t \odot h_{t-1})^T,$$

$$V_c \leftarrow V_c - \eta \sum_{t=1}^{T} (W_r(Be \odot (1 - z_t)) * (1 - c_t \odot c_t) x_t^T.$$

As in the "Vanilla" RNN, all the bias vectors are updated by omitting the outer product in the corresponding W or V update. The matrix B can also be a different random matrix for each parameter.

F. DFA Hyperparameter search

Hyperparameters are selected based on the best performances on a validation set among these possible values: hsize$\in$ [50,512], lr$\in$ [0.0005, 0.001,0.005,0.01], bs$\in$ [10,100,256], clip=2. The values selected by the model selection are:

1. Libras: Learning rate = 0.0005 (except for BPTT GRU: learning rate= 0.01), Hidden size = 512, Batch size = 10, Epochs= 900.
2. Strawberry: Learning rate = 0.0005 (except for BPTT GRU: learning rate= 0.005), Hidden size = 50 (except for RNN DFA: hidden size= 512), Batch size = 10 (except for RNN DFA: bs=100 and for RNN BPTT: bs= 256), Epochs= 300.
3. ECG200: [Learning rate = 0.0005 (Except for DFA GRU, lr=0.01), Hidden size = 50, Batch size = 256, Epochs= 500.
4. ROW-MNIST: [Learning rate=0.0005 (Except for RNN DFA and GRU DFA, lr=0.005), Hidden size = 512 (Except for RNN BPTT, hs= 50), Batch size = 100 (Except for RNN BPTT, bs = 10)].

In Figure 7 we show the learning curves of the test accuracy for the datasets ECG200 and Strawberry. The fact that the lines start at a different level is because the train, test, and validation sets are divided randomly so the test set can be particularly imbalanced. In these cases, the learning lines of DFA are not visibly growing. We believe that the restricted range of

European Innovation Council Funded by the European Union
Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

the hyperparameters prevented us to find solutions of DFA that work at best for these datasets.

G. Experiment setup for Lifelong Evolutionary Swarms

We use a population of 300 individuals and evolve the population for 200 generations on each task. NEAT starts with one hidden neuron and 50% of active connections, initialized randomly from a standard Gaussian distribution, and clamped between -5 and 5. NEAT adds or deletes a node with 20% probability, and mutates a weight by adding a random value sampled from a standard Gaussian with 80% probability.

We utilized the `neat-python` library (McIntyre et al. n.d.), which implements NEAT. The full configuration details are provided in Table 13. Notably, the compatibility threshold, which determines whether individuals belong to the same species based on genomic distance, is set to 3. Individuals with a genomic distance below this threshold are classified within the same species. We employed a modified sigmoidal transfer function, $\phi(x) = \frac{1}{1+e^{-4.9x}}$, as suggested in Kenneth O. Stanley and Risto Miikkulainen 2002. This steepened sigmoid improves precision at extreme activations and is nearly linear in the range between -0.5 and 0.5, optimizing its steepest ascent.

For a detailed description of all other configuration parameters, refer to the official documentation[7].

To determine the optimal regularization coefficient (λ), a systematic search was conducted. Since fitness values typically ranged between 20 and 30, we used this as a reference for the magnitude of penalization. An initial coarse search was performed with λ values of 5, 10, and 15. Based on the most promising results, we refined the search around 10 by experimenting with values 8, 9, 11, and 12.

To visualize the impact of regularization on NEAT's speciation mechanism, Figure 29 tracks the lifespan of each species, showing when they emerge and when they go extinct. With regularization (bottom), after the first task (200 generations) there are significantly fewer species, and they tend to persist longer. In contrast, the standard non-regularized evolution (top) exhibits a more dynamic speciation process, with species constantly appearing and disappearing.

The `SwarmForagingEnv` is a custom reinforcement learning environment designed for multi-agent swarm robotics tasks, where agents collaborate to retrieve specific target boxes in a dynamic and configurable environment. The environment supports episodic tasks and incorporates mechanisms for simulating task changes. The environment implements standard Gym API methods, including `step` and `reset`. Additionally, a `change_task` method enables updating the target color and the set of present colors, limited to those defined during initialization. In our experiments, each task features 2 unique colors, ensuring no overlap with colors from other tasks. For example, during the red task, boxes are red and blue, while during the green task, boxes are green and yellow. Table 14 details the initialization attributes of the class, including the values set for the experiments.

[7] `https://neat-python.readthedocs.io/en/latest/config_file.html`

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Table 13: NEAT Configuration Parameters

Parameter	Value
num_hidden	1
initial_connection	partial_direct 0.5
feed_forward	True
compatibility_disjoint_coefficient	1.0
compatibility_weight_coefficient	0.6
conn_add_prob	0.2
conn_delete_prob	0.2
node_add_prob	0.2
node_delete_prob	0.2
activation_default	neat_sigmoid
activation_options	neat_sigmoid
activation_mutate_rate	0.0
bias_init_mean	0.0
bias_init_stdev	1.0
bias_replace_rate	0.1
bias_mutate_rate	0.7
bias_mutate_power	0.5
bias_max_value	5.0
bias_min_value	-5.0
response_init_mean	1.0
response_init_stdev	0.0
response_replace_rate	0.0
response_mutate_rate	0.0
response_mutate_power	0.0
response_max_value	5.0
response_min_value	-5.0
weight_max_value	5
weight_min_value	-5
weight_init_mean	0.0
weight_init_stdev	1.0
weight_mutate_rate	0.8
weight_replace_rate	0.1
weight_mutate_power	1.0
enabled_default	True
enabled_mutate_rate	0.01
compatibility_threshold	3.0
species_fitness_func	max
max_stagnation	20
species_elitism	1
elitism	5
survival_threshold	0.2

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

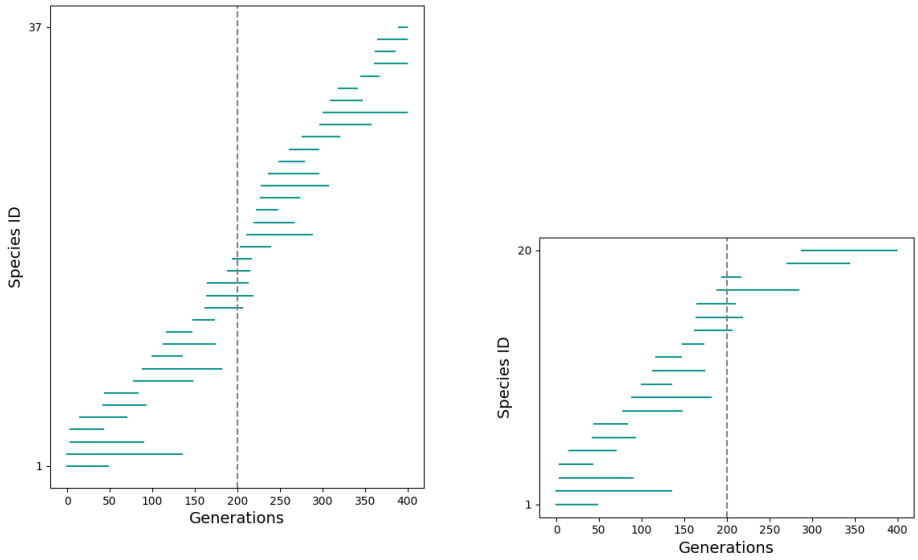


Figure 29: Lifespan of each species in evolution without regularization (top) and in evolution with regularization (bottom).

References

Bauer, F. L. and C. T. Fike (1960). “Norms and exclusion theorems”. In: *Numerische Mathematik* 2.1, pp. 137–141.

Carpenter, Gail and Stephen Grossberg (1986). “Adaptive Resonance Theory: Stable Self-Organization of Neural Recognition Codes in Response to Arbitrary Lists of Input Patterns”. In: *Proceedings of the Eight Annual Conference of the Cognitive Science Society*. Hillsdale, NJ: Erlbaum, pp. 45–62.

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (Oct. 1986). “Learning Representations by Back-Propagating Errors”. In: *Nature* 323.6088, pp. 533–536. ISSN: 1476-4687. DOI: 10.1038/323533a0. (Visited on 08/29/2024).

Reynolds, Craig W (1987). “Flocks, herds and schools: A distributed behavioral model”. In: *ACM SIGGRAPH Computer Graphics*. Vol. 21. 4. ACM, pp. 25–34.

Elman, Jeffrey L. (1990). “Finding Structure in Time”. In: *Cognitive Science* 14.2, pp. 179–211. ISSN: 1551-6709. DOI: 10.1207/s15516709cog1402_1. (Visited on 08/22/2019).

Werbos, P.J. (Oct. 1990). “Backpropagation through Time: What It Does and How to Do It”. In: *Proceedings of the IEEE* 78.10, pp. 1550–1560. ISSN: 1558-2256. DOI: 10.1109/5.58337. (Visited on 08/29/2024).

Caruana, Rich (July 1997). “Multitask Learning”. In: *Machine Learning* 28.1, pp. 41–75. ISSN: 1573-0565. DOI: 10.1023/A:1007379606734. (Visited on 10/14/2019).

Lohmiller, Winfried and Jean-Jacques E Slotine (1998). “On contraction analysis for non-linear systems”. In: *Automatica* 34.6, pp. 683–696.

French, Robert (Apr. 1999). “Catastrophic Forgetting in Connectionist Networks”. In: *Trends in Cognitive Sciences* 3.4, pp. 128–135. ISSN: 1364-6613, 1879-307X. DOI: 10.1016/S1364-6613(99)01294-2. (Visited on 08/14/2019).

Olszewski, Robert Thomas, Roy Maxion, and Dan Siewiorek (2001). “Generalized feature extraction for structural pattern recognition in time-series data”. PhD thesis. USA. ISBN: 0493538712.

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Table 14: `SwarmForagingEnv` class initialization parameters

Parameter Name	Value	Description
`size`	20 units	Size of the simulation arena, represented as a grid.
`target_color`	Varying	Target color for the boxes that agents must retrieve.
`n_agents`	5	Number of agents in the swarm.
`n_boxes`	20	Number of boxes in the arena.
`n_neighbors`	3	Number of nearest neighbors detected by an agent's sensors.
`sensor_range`	4 units	Maximum sensing range of agents.
`max_wheel_velocity`	2 units/s	Maximum wheel velocity for agent movement.
`sensitivity`	0.5 units	Threshold distance for agents to interact with a block.
`time_step`	0.1 s	Time step duration for each simulation step.
`duration`	500	Maximum number of steps per episode.
`max_retrieves`	20	Maximum number of boxes that can be retrieved in one episode.
`colors`	Varying	List of colors available for the boxes.
`season_colors`	Varying	List of colors available during a specific season.
`rate_target_block`	0.5	Proportion of target boxes among all boxes in the arena.
`repositioning`	`True`	Whether boxes are repositioned after being retrieved.
`efficency_reward`	`False`	Whether to reward agents for completing tasks before the maximum steps.
`see_other_agents`	`False`	Whether agents can detect other agents in the arena.
`boxes_in_line`	`False`	Whether boxes are placed in a line streight line during initialization.

Jaeger, Herbert (2002). *Tutorial on Training Recurrent Neural Networks, Covering BPPT, RTRL, and the "Echo State Network" Approach*. Tech. rep. GMD Report 159. Fraunhofer Institute for Autonomous Intelligent Systems.

Stanley, Kenneth O. and Risto Miikkulainen (2002). "Evolving Neural Networks through Augmenting Topologies". In: *Evolutionary Computation* 10.2, pp. 99–127.

Stanley, Kenneth Owen and Risto P. Miikkulainen (2004). "Efficient evolution of neural networks through complexification". AAI3143474. PhD thesis. The University of Texas at Austin. ISBN: 0496012517.

Stanley, K.O., B.D. Bryant, and R. Miikkulainen (2005). "Real-time neuroevolution in the NERO video game". In: *IEEE Transactions on Evolutionary Computation* 9.6, pp. 653–668. DOI: 10.1109/TEVC.2005.856210.

Kashtan, Nadav, Elad Noor, and Uri Alon (2007). "Varying environments can speed up evolution". In: *Proceedings of the National Academy of Sciences* 104.34, pp. 13711–13716. DOI: 10.1073/pnas.0611630104. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.0611630104. URL: https://www.pnas.org/doi/abs/10.1073/pnas.0611630104.

Şahin, Erol and Alan Winfield (2008). "Special issue on swarm robotics". In: *Swarm Intelligence* 2.2, pp. 69–72.

Dias Daniel, Peres Sarajane and Bscaro Helton (2009). *Libras Movement*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5GC82.

Hauert, Sabine, J-C Zufferey, and Dario Floreano (2009). "Reverse-engineering of artificially evolved controllers for swarms of robots". In: *IEEE Congress on Evolutionary Computation, CEC'09*. IEEE. Trondheim, Norway, pp. 55–61.

Lukoševičius, Mantas and Herbert Jaeger (Aug. 2009). "Reservoir Computing Approaches to Recurrent Neural Network Training". In: *Computer Science Review* 3.3, pp. 127–149. ISSN: 1574-0137. DOI: 10.1016/j.cosrev.2009.03.005. (Visited on 04/18/2020).

Winfield, Alan FT (2009). "Towards an engineering science of robot foraging". In: *9th International Symposium on Distributed Autonomous Robotic Systems (DARS 2008)*. Ed. by H Asama et al. Tsukuba, Japan: Springer, pp. 185–192.

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Lehman, Joel and Kenneth O. Stanley (June 2011). “Improving Evolvability through Novelty Search and Self-Adaptation”. In: *2011 IEEE Congress of Evolutionary Computation (CEC)*. New Orleans, LA, USA: IEEE, pp. 2693–2700. ISBN: 978-1-4244-7834-7. DOI: 10.1109/CEC.2011.5949955. (Visited on 03/18/2024).

Deng, Li (2012). “The mnist database of handwritten digit images for machine learning research”. In: *IEEE Signal Processing Magazine* 29.6, pp. 141–142.

Yildiz, Izzet B, Herbert Jaeger, and Stefan J Kiebel (2012). “Re-visiting the echo state property”. In: *Neural networks* 35, pp. 1–9.

Cho, Kyunghyun et al. (Oct. 2014). “On the Properties of Neural Machine Translation: Encoder–Decoder Approaches”. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, pp. 103–111. DOI: 10.3115/v1/W14-4012. (Visited on 09/21/2019).

Chung, Junyoung et al. (Dec. 2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. arXiv: 1412.3555 [cs]. (Visited on 09/05/2022).

Yao, Yao, Kathleen Marchal, and Yves Van de Peer (2014). “Improving the adaptability of simulated evolutionary swarm robots in dynamically changing environments”. In: *PLoS One* 9.3, e90695.

Ellefsen, Kai Olav, Jean-Baptiste Mouret, and Jeff Clune (Apr. 2015). “Neural Modularity Helps Organisms Evolve to Learn New Skills without Forgetting Old Skills”. In: *PLOS Computational Biology* 11.4, pp. 1–24. DOI: 10.1371/journal.pcbi.1004128. URL: https://doi.org/10.1371/journal.pcbi.1004128.

Hecker, Joshua P. and Melanie E. Moses (Mar. 2015). “Beyond pheromones: evolving error-tolerant, flexible, and scalable ant-inspired robot swarms”. In: *Swarm Intelligence* 9.1, pp. 43–70. ISSN: 1935-3820. DOI: 10.1007/s11721-015-0104-z. URL: https://doi.org/10.1007/s11721-015-0104-z.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (May 2015). “Deep Learning”. In: *Nature* 521.7553, pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539. (Visited on 09/03/2019).

Mouret, Jean-Baptiste and Jeff Clune (Apr. 2015). *Illuminating Search Spaces by Mapping Elites*. arXiv: 1504.04909 [cs, q-bio]. (Visited on 08/05/2024).

Silva, Fernando et al. (Sept. 2015). “odNEAT: An Algorithm for Decentralised Online Evolution of Robotic Controllers”. In: *Evolutionary Computation* 23.3, pp. 421–449. ISSN: 1063-6560. DOI: 10.1162/EVCO_a_00141. eprint: https://direct.mit.edu/evco/article-pdf/23/3/421/1518622/evco_a_00141.pdf. URL: https://doi.org/10.1162/EVCO%5C_a%5C_00141.

Castello, Eduardo et al. (2016). “Adaptive foraging for simulated and real robotic swarms: the dynamical response threshold approach”. In: *Swarm Intelligence* 10, pp. 1–31.

Francesca, Gianpiero and Mauro Birattari (2016). “Automatic Design of Robot Swarms: Achievements and Challenges”. In: *Frontiers in Robotics and AI* 3, p. 29.

Fricke, G. Matthew et al. (2016). “A distributed deterministic spiral search algorithm for swarms”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4430–4436. DOI: 10.1109/IROS.2016.7759652.

Lillicrap, Timothy P., Daniel Cownden, et al. (Nov. 2016). “Random Synaptic Feedback Weights Support Error Backpropagation for Deep Learning”. In: *Nature Communications* 7.1, pp. 1–10. ISSN: 2041-1723. DOI: 10.1038/ncomms13276. (Visited on 04/19/2020).

Mengistu, Henok, Joel Lehman, and Jeff Clune (July 2016). “Evolvability Search: Directly Selecting for Evolvability in Order to Study and Produce It”. In: *Proceedings of the Genetic and Evolutionary Computation Conference 2016*. GECCO ’16. New York, NY, USA: Association for Computing Machinery, pp. 141–148. ISBN: 978-1-4503-4206-3. DOI: 10.1145/2908812.2908838. (Visited on 03/17/2024).

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Nøkland, Arild (2016). “Direct Feedback Alignment Provides Learning in Deep Neural Networks”. In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee et al. Curran Associates, Inc., pp. 1037–1045.

Pugh, Justin K., Lisa B. Soros, and Kenneth O. Stanley (July 2016). “Quality Diversity: A New Frontier for Evolutionary Computation”. In: *Frontiers in Robotics and AI* 3. ISSN: 2296-9144. DOI: 10.3389/frobt.2016.00040. (Visited on 03/15/2024).

Chang, Shiyu et al. (2017). “Dilated recurrent neural networks”. In: *Advances in neural information processing systems* 30.

Ericksen, John, Melanie Moses, and Stephanie Forrest (2017). “Automatically evolving a general controller for robot swarms”. In: *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8. DOI: 10.1109/SSCI.2017.8285399.

Gallicchio, Claudio and Alessio Micheli (2017). “Echo state property of deep reservoir computing networks”. In: *Cognitive Computation* 9, pp. 337–350.

Inubushi, Masanobu and Kazuyuki Yoshimura (2017). “Reservoir computing beyond memory-nonlinearity trade-off”. In: *Scientific reports* 7.1, p. 10199.

Kirkpatrick, James et al. (Mar. 2017). “Overcoming Catastrophic Forgetting in Neural Networks”. In: *Proceedings of the National Academy of Sciences* 114.13, pp. 3521–3526. DOI: 10.1073/pnas.1611835114. (Visited on 11/22/2022).

Lake, Brenden M et al. (2017). “Building machines that learn and think like people”. In: *Behavioral and brain sciences* 40, e253.

Scellier, Benjamin and Yoshua Bengio (2017). “Equilibrium Propagation: Bridging the Gap between Energy-Based Models and Backpropagation”. In: *Frontiers in Computational Neuroscience* 11. ISSN: 1662-5188. (Visited on 11/16/2023).

Vaswani, Ashish et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30.

Zenke, Friedemann, Ben Poole, and Surya Ganguli (July 2017). “Continual Learning Through Synaptic Intelligence”. In: *International Conference on Machine Learning*, pp. 3987–3995. (Visited on 06/28/2019).

Bredeche, Nicolas, Evert Haasdijk, and Abraham Prieto (2018). “Embodied Evolution in Collective Robotics: A Review”. In: *Frontiers in Robotics and AI* 5, p. 12.

Chaudhry, Arslan et al. (2018). “Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence”. In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 556–572. ISBN: 978-3-030-01252-6. DOI: 10.1007/978-3-030-01252-6_33.

Birattari, Mauro et al. (2019). “Automatic Off-Line Design of Robot Swarms: A Manifesto”. In: *Frontiers in Robotics and AI* 6, p. 59.

Chang, Bo et al. (2019). “AntisymmetricRNN: A dynamical system view on recurrent neural networks”. In: *arXiv preprint arXiv:1902.09689*.

Crafton, Brian et al. (May 2019). “Direct Feedback Alignment With Sparse Connections for Local Learning”. In: *Frontiers in Neuroscience* 13. ISSN: 1662-453X. DOI: 10.3389/fnins.2019.00525. (Visited on 09/04/2024).

Ernoult, Maxence et al. (2019). “Updates of Equilibrium Prop Match Gradients of Backprop Through Time in an RNN with Static Input”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc. (Visited on 01/26/2024).

Goyal, Anirudh et al. (2019). “Recurrent independent mechanisms”. In: *arXiv:1909.10893*.

Jones, Simon, Alan FT Winfield, et al. (2019). “Onboard Evolution of Understandable Swarm Behaviors”. In: *Advanced Intelligent Systems* 1.

Parisi, German I. et al. (May 2019). “Continual Lifelong Learning with Neural Networks: A Review”. In: *Neural Networks* 113, pp. 54–71. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2019.01.012. (Visited on 12/28/2019).

European Innovation Council
Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Paszke, Adam et al. (2019). “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc.

Han, Donghyeon et al. (June 2020). *Extension of Direct Feedback Alignment to Convolutional and Recurrent Neural Network for Bio-plausible Deep Learning*. DOI: 10.48550/arXiv.2006.12830. arXiv: 2006.12830 [cs, stat]. (Visited on 08/28/2024).

Launay, Julien et al. (Dec. 2020). *Hardware Beyond Backpropagation: A Photonic Co-Processor for Direct Feedback Alignment*. DOI: 10.48550/arXiv.2012.06373. arXiv: 2012.06373 [cs, stat]. (Visited on 09/04/2024).

Lesort, Timothée et al. (June 2020). “Continual Learning for Robotics: Definition, Framework, Learning Strategies, Opportunities and Challenges”. In: *Information Fusion* 58, pp. 52–68. ISSN: 1566-2535. DOI: 10.1016/j.inffus.2019.12.004. (Visited on 05/06/2020).

Lillicrap, Timothy P., Adam Santoro, et al. (Apr. 2020). “Backpropagation and the Brain”. In: *Nature Reviews Neuroscience*, pp. 1–12. ISSN: 1471-0048. DOI: 10.1038/s41583-020-0277-3. (Visited on 04/18/2020).

Schranz, Melanie et al. (2020). “Swarm Robotic Behaviors and Current Applications”. In: *Frontiers in Robotics and AI* 7, p. 36.

Cossu, Andrea, Davide Bacciu, et al. (2021). “Continual Learning with Echo State Networks”. In: *ESANN 2021 Proceedings*. Online event (Bruges, Belgium): Ciaco - i6doc.com, pp. 275–280. ISBN: 978-2-87587-082-7. DOI: 10.14428/esann/2021.ES2021-80. (Visited on 12/19/2023).

Dorigo, Marco, Guy Theraulaz, and Vito Trianni (2021). “Swarm robotics: Past, present, and future [point of view]”. In: *Proceedings of the IEEE* 109.7, pp. 1152–1165.

Hayes, Tyler L. et al. (Oct. 2021). “Replay in Deep Learning: Current Approaches and Missing Biological Elements”. In: *Neural computation* 33.11, pp. 2908–2950. ISSN: 0899-7667. DOI: 10.1162/neco_a_01433. (Visited on 08/30/2023).

Cossu, Andrea, Gabriele Graffieti, et al. (2022). “Is Class-Incremental Enough for Continual Learning?” In: *Frontiers in Artificial Intelligence* 5. ISSN: 2624-8212. DOI: 10.3389/frai.2022.829842. (Visited on 03/24/2022).

Filipovich, Matthew J. et al. (Dec. 2022). “Silicon Photonic Architecture for Training Deep Neural Networks with Direct Feedback Alignment”. In: *Optica* 9.12, pp. 1323–1332. ISSN: 2334-2536. DOI: 10.1364/OPTICA.475493. (Visited on 09/04/2024).

Jones, Simon, Emma Milner, et al. (2022). *DOTS: An Open Testbed for Industrial Swarm Robotic Solutions*. arXiv: 2203.13809 [cs.RO]. URL: https://arxiv.org/abs/2203.13809.

Khetarpal, Khimya et al. (2022). “Towards Continual Reinforcement Learning: A Review and Perspectives”. In: *Journal of Artificial Intelligence Research* 75, pp. 1401–1476. ISSN: 1076-9757. DOI: 10.1613/jair.1.13673.

Kozachkov, Leo, Michaela Ennis, and Jean-Jacques Slotine (2022). “RNNs of RNNs: Recursive construction of stable assemblies of recurrent neural networks”. In: *Advances in neural information processing systems* 35, pp. 30512–30527.

Merlin, G. et al. (2022). “Practical Recommendations for Replay-Based Continual Learning Methods”. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 13374 LNCS, pp. 548–559. ISSN: 0302-9743. DOI: 10.1007/978-3-031-13324-4_47.

Nakajima, Mitsumasa et al. (Dec. 2022). “Physical Deep Learning with Biologically Inspired Training Method: Gradient-Free Approach for Physical Hardware”. In: *Nature Communications* 13.1, p. 7847. ISSN: 2041-1723. DOI: 10.1038/s41467-022-35216-2. (Visited on 08/28/2024).

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.

Wright, Logan G. et al. (Jan. 2022). “Deep Physical Neural Networks Trained with Backpropagation”. In: *Nature* 601.7894, pp. 549–555. ISSN: 1476-4687. DOI: 10.1038/s41586-021-04223-6. (Visited on 08/29/2024).

Hemati, Hamed et al. (Nov. 2023). “Class-Incremental Learning with Repetition”. In: *Proceedings of The 2nd Conference on Lifelong Learning Agents*. PMLR, pp. 437–455. (Visited on 12/19/2023).

Masana, Marc et al. (May 2023). “Class-Incremental Learning: Survey and Performance Evaluation on Image Classification”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.5, pp. 5513–5533. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2022.3213473. (Visited on 06/06/2024).

Momeni, Ali et al. (Dec. 2023). “Backpropagation-Free Training of Deep Physical Neural Networks”. In: *Science* 382.6676, pp. 1297–1303. DOI: 10.1126/science.adi8474. (Visited on 08/29/2024).

Giannini, Federico et al. (2024). “Streaming Continual Learning for Unified Adaptive Intelligence in Dynamic Environments”. In: *IEEE Intelligent Systems* 39.6, pp. 81–85. ISSN: 1941-1294. DOI: 10.1109/MIS.2024.3479469.

Sinha, Sania, Tanawan Premsri, and Parisa Kordjamshidi (2024). “A Survey on Compositional Learning of AI Models: Theoretical and Experimetnal Practices”. In: *arXiv preprint arXiv:2406.08787*.

Towers, Mark et al. (Nov. 2024). *Gymnasium: A Standard Interface for Reinforcement Learning Environments*. DOI: 10.48550/arXiv.2407.17032. arXiv: 2407.17032 [cs]. (Visited on 01/09/2025).

Verwimp, Eli et al. (2024). “Continual Learning: Applications and the Road Forward”. In: *Transactions on Machine Learning Research*. ISSN: 2835-8856.

K. Kemsley, A. Bagnall (n.d.). *Strawberry*. https://timeseriesclassification.com.

McIntyre, Alan et al. (n.d.). *neat-python*. URL: https://neat-python.readthedocs.io/en/latest/.

European Innovation Council Funded by the European Union

Funded by the European Union under Grant Agreement 101070918. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Innovation Council and SMEs Executive Agency (EISMEA). Neither the European Union nor the granting authority can be held responsible for them.