IMPERIAL

PHYSICAL CONSTRAINTS AND FUNCTIONAL DEMANDS SHAPE MODULAR NEUROMORPHIC INTELLIGENCE

GABRIEL BÉNA

Thesis submitted for the degree of Doctor of Philosophy

Supervised by DAN F.M GOODMAN

Electrical and Electronical Engineering
Imperial College London

March 26, 2026

COPYRIGHT DECLARATION

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution Non-Commercial No Derivatives license (CC BY-NC-ND).

Under this licence, you may copy and redistribute the material in any medium or format on the condition that; you credit the author, do not use it for commercial purposes and do not distribute modified versions of the work.

When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

ORIGINALITY DECLARATION

I hereby declare that this thesis and the work herein detailed, was composed and originated by myself, except where appropriately referenced and credited.

Montpellier, February 13, 2026

Gabriel Béna

ABSTRACT

Modularity—the decomposition of complex systems into semi-autonomous, reusable parts—is widely regarded as a foundational organising principle of intelligence. Yet the concept remains elusive: definitions shift across fields, and the causal mechanisms linking structure to function are poorly understood. This thesis investigates modularity in neural networks from multiple angles, and in doing so, traces an arc from studying it as a measurable property to building self-organising substrates where it could emerge. We first disentangle modularity along two axes: its causal locus (physical vs. functional) and its causal role (imposed vs. emergent). Using this framework, we show that imposing structural modularity yields functional specialisation only under specific resource constraints and environmental separability. We then demonstrate that enforcing the metabolic costs of long-range connectivity via spatial embedding naturally produces sparse, modular topologies. These structurally modular networks outperform unstructured architectures on compositional tasks, proving physical wiring costs can drive functionally beneficial modularity. Pivoting to self-organisation, we demonstrate that locally-connected Neural Cellular Automata can master a wide range of computational primitives like matrix operations, and support neural network emulation. Next, we introduce a scale-free framework for self-organising digital circuits. Here, a topology-masked Transformer replaces global back-propagation with local message passing to act as a decentralised meta-optimizer. This system self-assembles functional Boolean circuits, maintains homeostasis, and dynamically re-routes logic around permanent hardware faults—exhibiting adaptive resilience that generalises from small to large circuits without retraining. Together, these investigations reveal modularity not as a single property to engineer, but as an emergent phenomenon—shaped by physical constraints, driven by functional demands, and mediated by local, self-organising rules. They suggest that the path toward robust, scalable intelligence may lie in cultivating substrates that grow, learn, and repair themselves from the ground up.

REMERCIEMENTS

To *Dan*, the best PhD supervisor I could have ever hoped for. Thank you for always letting me pursue what got me excited and passoniate, even when that meant taking the time for me to find out what that was. Thank you for being always so kind, invested and filled with a true sense of wonder in the beauties of science. Thank you for letting me take *les grandes vacances*, and for always pushing me to put my physical and mental health first and above all else. Thank you for showing me how we could, and why we should, do better science, by being more open, honest and self-organised, moving away from predatory systems that profit from the many, and only benefit the few. Thanks to everyone in the Neural Reckoning lab over the years. To Marcus for being the patient, inspiring, groovy, "dammit-he-seems-like-he's-got-it-all-figured-out" person that he indeed is. All the future students of Pr. Ghosh are lucky bastards. Finally, special thanks to my jury: Jack, Emre, and Jeremy. Thank you so much for your time. I am very proud of this thesis, and I hope you have as much fun reading it as I had writing it!

Thanks to the Neuromorphic community and to Capo-Caccia workshop pals: Jimmy, Gabriel, Marcello, Balázs, Mahmoud, Melika, Emre (again!), Mihai, Toby, and so many more. I'm so grateful to have found myself back in this niche, full of fascinating people, and stories to explore. As a bonus, you get to spend a couple of weeks per year in beautiful Sardinia, listening to the most inspiring researchers during the day, snorkelling or playing volleyball at twilight, then frantically working on some mad scientist project during the night while drinking stolen wine and blasting some King Gizzard. A special gratitude extended to Melika, for the trust invested in me to invite me to INI, and to all the exciting research that's still ahead of us !

To London, the city that keeps on giving, filled to the brim with the coolest, quirkiest, hippest and most intriguing places, bars, venues and concepts you could imagine. Thanks to all the amazing artists I was lucky enough to see, and to all the partying I was able to have, in this one of a kind cultural epicentre: Ezra Collective, Little Simz, Crooksie, Kendrick, Iseo & Dodosound, and so so many more. To the musicals, and to the immersive theatres. To FOLD, Corsica Studios, E1, the French-Press parties and all the places where we danced and forgot ourselves in the music and the crowd. To the rugby evenings, the pubs and the hour-long journeys home. To the peaceful West, and to the frantic East. To Ronan, Michael, and this horrible horrible first flat in which we had so much fun. To De-Morgan House, and to my lovely roommates there, to Harry and to Fran, to Calvin and to our sunny mornings on the patio, to the games at the Stoop, to the barbecues and the Sundays playing ball in the park. To Hammers rugby club, both the rainy muddy games, and the sunny French-flairy ones. To Wapping, a haven in the east, to Genesis cinema and the bar-thrash Wednesdays, to Hackney and Shoreditch, Cinnamon

Cafe and the Captain Kidd. To padel-fever, to the Stratford grand-Prix and all the rage and bliss it induced. To all the Gs that we split, and all the ones that we missed.

Merci à tous mes amis, d'être de super copains, des humains beaux et généreux que j'ai appris à connaître et respecter. Merci à Elliott, à Stumi, et à Grégoire, à Taupi et à Corberes, cette belle équipe londonienne qui ont fait de ce périple outre-manche un plaisir partagé. À tous mes amis d'école, Théo, mon agapè de Montpellier, Ludo et Titouan, mes compères sur le sol, sur la neige, et peut-être un jour dans les airs, Orion et Louise, ma paire d'humain favorite, avec qui je rêve de construire un monde meilleur, Arthur notre petit bouddha de poche, que j'arriverai un jour à convertir au samsara, Oliv et Maelle, les pichnous chez qui j'ai toujours eu un lit douillet disponible, accompagné d'offres de soirées que je n'ai pu refuser, Sylvain, avec qui je reste ami en espérant qu'il devienne la rockstar qu'il mérite tant être, Clara, la mère castors aux mille histoires et la battante aux mille épreuves et Mathou qui cherche sa voie et m'inspire dans sa bataille pour la paix intérieure, amourachées qui nous ont rejoints dans l'aventure... 10 ans déjà! Merci à Maxence d'avoir été mon ami, mon co-auteur ainsi qu'une vraie inspiration intellectuelle à travers la seconde moitié de ce manuscrit. Aux amis de prépa que j'aimerais revoir un peu plus et à toutes ces personnes que j'ai rencontrées dans ma vie et dans ce monde, du Japon, à la Colombie, de Toulouse à Londres, de Paris à Montpellier. Bientôt Arthur-ville? Je vous aime tous.

Merci à ma famille pour m'avoir toujours soutenu et aimé. À mon père, papa-poule, pour m'avoir hébergé, nourri et chouchouté pendant la phase finale d'écriture, pour les jus d'orange du matin, pour m'avoir guidé dans ma carrière scientifique, et pour m'avoir toujours poussé à faire mieux, à partir en prépa, à croire en moi. À ma mère, pour tous ses conseils précieux, pour être une inspiration constante d'utopisme et d'imaginaires, pour nos pauses-café-discussions qui s'éternisent le midi sous le soleil, pour me donner envie de me battre et de prendre la voie de l'espoir, pour me montrer qu'il n'y a pas d'âge pour entamer de nouveaux chemins de vie. À mon frère et à ma sœur, ma triforce que j'aime si fort. À nos voyages tous les trois, et à tous ceux qui suivront encore. À Jonas pour m'avoir fait plein de petits plats et de chocs mentaux pendant notre colocation de ces derniers mois, pour ta gentillesse, pour toutes les musiques que tu m'as fait découvrir et pour toutes nos sessions cuisine + Dire Straits. À Éloane, enfant du soleil, qui me stupéfait toujours par le savant mélange de sagesse et d'énergie enfantine qu'elle dégage, qui m'inspire beaucoup malgré son jeune âge, allons danser ensemble bientôt. À Laura, avec qui j'ai partagé une partie de ce chemin, pour qui la thèse a commencé de manière tellement plus dure que la mienne, mais qui, par sa force d'âme et grâce à une belle équipe finale, aperçoit également le bout du tunnel ! À Montpellier, ma ville que j'aime tant, et où j'espère vivre un jour. À Gagny et à Mamie-Josée, qui m'ont donné les forces de traverser la prépa. À la Corse, à Belgodère et à mes grand-parents adorés, chez qui je n'ai pas tant écrit, mais où j'ai toujours trouvé des ressources inespérées. À Issa, à jamais dans nos cœurs. À Majo pour sa tendresse et sa joie. À Martin, pour m'avoir montré les étoiles et parlé de la Vie, amorçant ainsi le chemin qui m'a mené jusqu'ici.

This thesis is dedicated to the first human who's ever looked up at the stars, or down at the earth, and wondered what other beautiful worlds were hiding in plain sight...

CONTENTS

GENERAL INTRODUCTION 19

0.1 The Grand Puzzle: Emergence of Intelligence . . . 19
0.2 Modularity: A Story from Two Perspectives . . . 20
0.2.1 Force 1: "From Below" – Physical and Proximate Causes . . . 21
0.2.2 Force 2: "From Above" – Functional and Ultimate Causes . . . 21
0.3 A Causal Ambiguity? . . . 22
0.4 Pressures vs. Signatures . . . 22
0.5 State of the Art: Brain Science (Re-illuminated) . . . 22
0.5.1 The Common Ancestor: Strict Localization ($\mathsf{PS} \equiv \mathsf{US}$) . . . 23
0.5.2 The Holist Counter-Movement ($\mathsf{PS} \neq \mathsf{US}$) . . . 24
0.5.3 The Neo-Faculty School ($\mathsf{UP} \rightarrow \mathsf{US}$) . . . 25
0.5.4 A Modern but Partial Synthesis: Network Neuroscience ($\mathsf{PS} \overset{?}{\leftrightarrow} \mathsf{US}$) . . . 26
0.5.5 The Conflation of Property and Purpose . . . 27
0.5.6 Retaining Causality: Lesion Analysis ($\mathsf{PP} \rightarrow \mathsf{US}$) . . . 27
0.5.7 The Brain Economy Principle ($\mathsf{PP} \rightarrow \mathsf{PS}$) . . . 28
0.5.8 Everything, Everywhere, all at Once ? . . . 28
0.6 The Neuro-AI Opportunity: From Observation to Causation . . . 29
0.6.1 Mapping Experiments, Not Just Concepts . . . 30
0.6.2 The Experimenter as "Evolution Short-Circuiter" . . . 30
0.7 State of the Art: The Computational Exploration . . . 31
0.7.1 Route 1: The Architects . . . 31
0.7.2 Route 2: The Emergentists . . . 33
0.8 Roadmap of the Thesis . . . 38

I INTELLIGENCE UNDER CONSTRAINTS: RESOURCES, SPACE, AND MODULARITY

1 DYNAMICS OF SPECIALIZATION IN NEURAL MODULES UNDER RESOURCE CONSTRAINTS 42

1.1 Abstract . . . 44
1.2 Introduction . . . 44
1.3 Methods . . . 48
1.3.1 Environment: Data and Tasks . . . 48
1.3.2 Networks . . . 49
1.3.3 Structural modularity . . . 50
1.3.4 Metrics . . . 51

1.4 Results . . . 52
1.4.1 Moderate structural modularity is not a sufficient condition for specialization . . . 53
1.4.2 Environmental structure and resource constraints determine specialization 54
1.4.3 Function specialization can vary dynamically . . . 57
1.5 Discussion . . . 58

2 SPATIAL AND NEUROMORPHIC PRIORS FOR COMPOSITIONAL LEARNING 63
2.1 Introduction . . . 66
2.1.1 The Combinatorial Imperative: From Monolithic Mapping to Compositional Reasoning . . . 66
2.1.2 The Landscape of Compositional Generalization . . . 67
2.1.3 The Mechanistic Gap: Controlling Complexity and Emergence . . . 68
2.1.4 Biological Constraints as Inductive Biases . . . 69
2.1.5 Contributions and Key Findings . . . 69
2.2 Methods . . . 70
2.2.1 The 2D-PVR Framework: A Tunable Compositional Testbed . . . 70
2.2.2 Spatial DEEP-R: Hardware-Software Co-Design . . . 75
2.2.3 Network Architecture . . . 78
2.2.4 Statistical Framework: Matched-Pairs Analysis . . . 79
2.2.5 Functional Analysis: The Selectivity-Clustering-Lesion Pipeline . . . 81
2.3 Results . . . 83
2.3.1 Sanity Check: Static vs Dynamic optimization . . . 83
2.3.2 Task Difficulty and Learning Dynamics . . . 83
2.3.3 Enhanced Generalization Under Spatial Constraints . . . 84
2.3.4 Structural Modularity Emergence . . . 85
2.3.5 Width-Dependent Effects . . . 86
2.3.6 The Structure-Function Gap: Modularity without Specialization . . . 88
2.4 Conclusion . . . 89

II SELF-ORGANISING PRINCIPLES OF INTELLIGENCE

3 A PATH TO UNIVERSAL NEURAL CELLULAR AUTOMATA 92
3.1 Abstract . . . 94
3.2 Introduction . . . 94
3.3 Related Work . . . 96
3.4 Methods . . . 98
3.4.1 General Setup . . . 98
3.4.2 Neural Cellular Automata . . . 99
3.4.3 Hardware (Immutable State) . . . 100

3.4.4 Tasks . . . 102
3.4.5 Training . . . 103
3.5 Experiments and Results . . . 103
3.5.1 Task Training . . . 104
3.5.2 MNIST Classifier emulation . . . 104
3.5.3 Future directions: task composition and neural compiler . . . 106
3.6 Conclusion . . . 109

4 SELF-ORGANISING DIGITAL CIRCUITS 111
4.1 Abstract . . . 114
4.2 Introduction . . . 114
4.3 Methods . . . 117
4.3.1 Differentiable Logic Gate Networks . . . 117
4.3.2 Graph Representation of Circuits . . . 120
4.3.3 Meta-Learning Architecture: Topology-Masked Transformer . . . 120
4.3.4 Training Framework: Pool-Based Meta-Learning . . . 123
4.4 Experiments and Results . . . 124
4.4.1 Regime I: Validation on Fixed Topologies . . . 125
4.4.2 Regime II: Generalisation to Random Topologies . . . 128
4.4.3 Regime III: Scale-Free Optimisation . . . 129
4.5 Discussion . . . 130

GENERAL DISCUSSION AND CONCLUSION 133

A APPENDIX FOR CHAPTER 1 139
A.1 Q measure derivation . . . 139
A.2 Network Accuracy . . . 140
A.3 Decision Dynamics . . . 140
A.4 Parameter sweeps for structure-function relationship . . . 141
A.5 Code examples . . . 141
A.5.1 Noisy inputs . . . 141
A.5.2 Dynamic inputs . . . 142

B APPENDIX FOR CHAPTER 2 145

AUTHOR'S LIST OF PUBLICATIONS 149

BIBLIOGRAPHY 150

LIST OF FIGURES

Figure 0.1 A small history of neuroscience research 24
Figure 0.2 Studies and usages of emerging modularity in AI research. 31
Figure 1.1 Methods Summary 53
Figure 1.2 Levels of functional specialization in networks 55
Figure 1.3 Functional specialization heatmap of networks 57
Figure 1.4 Specialization dynamics 59
Figure 2.1 Baseline: Linear Chaining 72
Figure 2.2 Original PVR and C-PVR Datasets 73
Figure 2.3 Schematic representation of the 2D Grid 74
Figure 2.4 Illustration of the Mosaic neuromorphic architecture 78
Figure 2.5 Spatial Penalty matrices 79
Figure 2.6 Summary Figure of the architecture and data flow 80
Figure 2.7 Training and "lift-off" dynamics 84
Figure 2.8 Spatial-embeddings net effects on multiple metrics 86
Figure 2.9 Spatial structure emerging in sparse embedded networks 87
Figure 2.10 Effect of spatial embeddings on generalization at different width 87
Figure 2.11 Specialization Results: Functional and Structural Task and Module Entropy 89
Figure 3.1 Framework Teaser Figure 94
Figure 3.2 Schematic of our architecture, showing the distinction between mutable (computational) and immutable (hardware) states. 99
Figure 3.3 Monolithic hardware configurations for 3 different sub-tasks 101
Figure 3.4 Modular hardware configurations for 3 different sub-tasks, and (now varying) different matrix sizes and placement. 101
Figure 3.5 NCA Emulation of an entire neural-network 105
Figure 3.6 Out Of Distribution Tasks 106
Figure 3.7 Composite Tasks Chaining 107
Figure 3.8 Graph-based tasks representations and GNN-based hypernetwork for hardware generation 108
Figure 4.1 Circuit Architecture and Graph Representation 121
Figure 4.2 Summary of training procedure 124
Figure 4.3 NCA Performance on Fixed Topologies. 126
Figure 4.4 NCA Resilience to Damages. 127
Figure 4.5 PCA Trajectories of Circuit Optimization 129
Figure 4.6 NCA Performance on Random Topologies 130

Figure 4.7 Scale-Free Generalization 131
Figure A.1 Appendix: Accuracy of training 141
Figure A.2 Appendix: Decision-making dynamics over training 142
Figure A.3 Appendix: Specialization for all architectures and data. 143
Figure B.1 Appendix Chapter 2: General Accuracy Results 145
Figure B.2 Appendix Chapter 2: OOD generalization changes per L1 146
Figure B.3 Average evolution over training of the Q metric over training 146
Figure B.4 Specialization Analysis Example 147

LIST OF TABLES

Table 0.1 The Four Quadrants of Modularity Research 23
Table 4.1 Standard Look-Up Tables (Truth Tables) for 2-Input Logic Gates 118

ACRONYMS

PP	Proximate Pressure
UP	Ultimate Pressure
PS	Proximate Signature
US	Ultimate Signature
CA	Cellular Automata
NCA	Neural Cellular Automata
NDP	Neural Developmental Programs
LUT	Look-up Table
LGN	Logical Gate Network

GENERAL INTRODUCTION

0.1 THE GRAND PUZZLE: EMERGENCE OF INTELLIGENCE

Emergence stands as one of the most fundamental organizing principles in our universe. It describes the phenomenon whereby simple components, governed by local interactions, self-organize to form complex systems exhibiting behaviours far richer than the sum of their parts. These emergent properties are absent in the individual components; they exist only within the collective dynamics of the whole. However, this complexity is rarely unstructured. Instead, the organization of complexity usually implies the notion of **Modularity** (or near-decomposability), the tendency of systems to be composed of distinct, semi-autonomous units, and usually the notion of **Hierarchy** as well, where these modules can be nested across scales (Simon, 1962). It has been suggested that this is, fundamentally, what enables complex systems to even exist (Kirschner et al., 2005). As Herbert Simon famously illustrated with his "Watchmaker Argument," stable sub-assemblies enable longer, more complicated overall structures. If a system must be built from scratch every time it is disturbed, it will never reach high complexity. However, if stable intermediate states (modules) exist, the system achieves a robustness that allows it to persist and evolve. Simon went so far as to even posit that without such a fundamental organizational property of the world we live in, we would have been unable to understand much about it at all, and that in this sense "fortune smiled upon Kepler and Newton" (Konrad-Lorenz-Institute, 2005; Simon, 1977).

Thus, modularity appears to be a near-ubiquitous property of observable complex systems, and moreover not merely a convenient lens for observers but an almost fundamental necessity for the systems themselves, observed at every level (Konrad-Lorenz-Institute, 2005; Newman, 2006; Sauro, 2008; Simon, 1962; Solé et al., 2002; Watts and Strogatz, 1998). In physics, we observe the hierarchical clustering of particles into atoms, atoms into molecules, and molecules into macroscopic matter. In our societies, individuals form families, which aggregate into communities and nations. The example are endless, and the pattern persistent: units tend to form clusters, with which they connect more preferentially than with the rest. Biology offers perhaps the richest evidence for modularity, when excluding neural systems from the conversation. From the molecular to the organismal scale, biological organization is fundamentally modular. We can furthermore distinguishes between multiple types of modularities in biology: *morphological* (the structure, or function, of parts of organisms) and *developmental or evolutionary* (how processes of development and evolution themselves display modular organizations). In Systems Biology, we observe this in metabolic networks, protein-protein interaction maps,

and many more (Wagner et al., 2005). This structural compartmentalization is not accidental: Evolutionary and Developmental Biology (evo-devo) suggests it is a prerequisite for adaptation. Modularity would allow different parts of an organism (like a limb or a specific metabolic pathway) to evolve independently without disastrously interfering with other vital functions (Wagner et al., 2007). Modularity would, in this case, be found in the genotype to phenotype mapping, allowing a somewhat precise set of features to be influenced by a set of genes, without suffering much if the rest of the genome was disturbed, thus creating evolutionary building blocks (reminiscent of Simon's watchmaker argument).

Because of these fundamental observations, and because modularity aligns so naturally with the way we think and understand complexity, it has largely pervaded our design and engineering principles as well (Baldwin and Clark, 2000; Lipson, 2007; Marr, 1976; Suh, 1990). From the interchangeability of physical parts in industrial manufacturing to the principles of encapsulation and object-oriented programming in computer science, we successfully engineer complexity by dividing it into manageable, independent sub-problems. Finally, as we will investigate in depth, modularity is equally important a notion when it comes to studying the brain and biological neural networks. Since Artificial Neural Networks (ANNs) exist at the intersection of these two worlds (born from *biological* inspiration yet constructed through *engineering* principles) it is no surprise that the concept has permeated the field of *artificial* intelligence as well. As such, we will explore modularity as the principal and intuitive lens through which to understand and design artificial learning systems. But even if modularity *feels* intuitive and *looks* ubiquitous, do we really know what it means? It is a term often used but rarely rigorously defined, before we can use it, we must define it.

0.2 MODULARITY: A STORY FROM TWO PERSPECTIVES

Modularity pervades research, engineering, and arguably even art and culture (Konrad-Lorenz-Institute, 2005). This makes it a very *potent*, general, and unifying concept. However, it simultaneously makes it a very *loaded* concept. The literature reveals a landscape where definitions shift depending on the observer's field, creating a muddied picture when studying it as a scientific variable.

We argue that modularity in neural-networks is not a single, monolithic concept. Instead, it is the *result of an interplay* between two fundamental forces, or perspectives, that shape this notion.

0.2.1 *Force 1: "From Below" – Physical and Proximate Causes*

First, modularity is shaped by *low-level physical constraints*. Whether running on a biological brain, or a neuromorphic chip, neural networks are not abstract computational graphs; they are spatially embedded systems:

- **Metabolic Costs:** Systems possess finite energy budgets to build connections and maintain communication between processing elements.
- **Noisy Substrates:** Processing elements are rarely perfect; the system must be robust to internal signal degradation.
- **The Genomic Bottleneck:** Biological networks do not start as a *tabula rasa*. They must instead *grow* based on compressed information passed down through genes.
- **Local Plasticity:** Learning rules are often restricted to local information availability rather than global optimization.

We are talking here about a mechanistic perspective: the *how* of the system. We can relate this to the concept of **Proximate Causes** from Mayr (1961): the immediate mechanistic explanation of a trait.

0.2.2 *Force 2: "From Above" – Functional and Ultimate Causes*

Second, modularity is driven by *high-level functional pressures*. Neural networks do not simply exist for their own sake; they function as the control systems for agents acting within an environment. These agents are subject to evolutionary or adaptive pressures to survive and perform. Consequently, the network must solve complex problems to ensure the agent's persistence:

- **Compositionality:** The natural world is compositional, and is too complex to be memorized, or brute-forced. To navigate it effectively, agents must be able to decompose complex tasks into reusable sub-routines.
- **Generalization and Transfer Learning:** Environments are rarely identical. Agents must apply past experiences to novel situations to avoid fatal errors in unseen contexts.
- **Continual Learning for Adaptation:** The world is dynamic. Agents must adapt to new situations without overwriting the skills required for previous ones (avoiding catastrophic interference).

This is the adaptive perspective: the "why" of the system. We can relate this to the concept of **Ultimate Causes** from Mayr (1961): the evolutionary or adaptive explanation of a trait.

0.3 A CAUSAL AMBIGUITY?

This dual-force perspective offers a strong disentanglement of the concept. But looking closer, we spot a second knot that needs untying: a problem of *causal ambiguity*. For both the Low-Level and High-Level perspectives, the core concepts are used inconsistently as both *causes* and *effects*:

- Is "connection cost" (low-Level) a **prior** we impose on a model, or is "energy efficiency" a **result** we measure?
- Is "task compositionality" (high-Level) a **structure** we impose on the environment, or is "compositional reasoning" the **resulting organization** of the models we observe?

This confusion between modularity-as-a-driver (explanans) and modularity-as-a-result (explanandum) has fragmented the field (as noted in the introduction of Konrad-Lorenz-Institute, 2005 as well). To make progress, we must explicitly disambiguate these roles.

0.4 PRESSURES VS. SIGNATURES

To resolve this ambiguity, we propose a 2×2 framework that separates the *type* of cause from its *temporal role* in an experimental process.

- **The Primary Axis (Y-Axis): The Locus of Causation.** This is our original distinction from section 0.2:
  - Low-Level (Proximate) | Physical
  - High-Level (Ultimate) | Functional
- **The Secondary Axis (X-Axis): The Causal Role.** This is our new, clarifying distinction:
  - **Causal Pressures (Inputs):** The "drivers," "priors," or "constraints." These are the forces we (or the environment) impose on the system *before or during* the experiment.
  - **Emergent Signatures (Outputs):** The "results," "metrics," or "properties." These are the phenomena we can *measure* or *extract* from the system *after* the experiment.

This grid (visualized in table 0.1) creates four distinct quadrants, useful to understand research on modularity.

0.5 STATE OF THE ART: BRAIN SCIENCE (RE-ILLUMINATED)

Some of the research in neuroscience can be re-interpreted as a persistent, evolving search for the links within this framework. Critically, the quest to link structure to function, specifically their modular *signatures*, has been at the core of neuroscience for decades. As research has

Table 0.1: The Four Quadrants of Modularity Research

	Low-Level (Proximate) *(The "How" - Physical)*	**High-Level (Ultimate)** *(The "Why" - Functional)*
Causal Pressures or Priors *(Inputs)*	**Proximate Pressures (PPs)** *(e.g., Connection Cost, Noise, Energy Budget)*	**Ultimate Pressures (UPs)** *(e.g., Compositional Tasks, Varying Environments)*
Emergent Signatures *(Outputs)*	**Proximate Signatures (PSs)** *(e.g., Q-Metric, Small-Worldness, Structural Clustering)*	**Ultimate Signatures (USs)** *(e.g., Functional Specialization, Generalization, Neural Reuse)*

advanced, the very definitions of the quadrants have evolved, illustrating that these "signatures" are not static truths but properties that must be viewed through changing technological and conceptual lenses.

0.5.1 *The Common Ancestor: Strict Localization* (PS ≡ US)

Historically, the field was dominated by a strong *localizationist* perspective. In this case, the signatures investigated were clear: clearly defined structural modules with little or no emphasis on more complicated inter-modules connectivity, and a narrow definition of functional specialization. In this view, the mapping between structure and function was assumed to be bijective: one structural unit corresponds to one functional faculty.

We can trace back this doctrine, sometime called *Faculty Psychology*, back to Gall's phrenology era. While severely flawed in its methodology, Gall still made a number of lasting contributions that should not (as passing time does too often) be forgotten: the mind is composed of distinct functional faculties (**US**) and that these are housed in distinct physical organs (**PS**) (whose volume Gall erroneously speculated would create detectable variations in the skull). This view became more dominant and scientifically legitimized when Paul Broca, in 1861, studied the curious case of "Mr. Tan" (Broca, 1861) : This patient suffered damage to the frontal-lobe following a stroke which left him unable to pronounce any but one syllable: "Tan", while amazingly still being able to *comprehend* language. Shortly thereafter, Wernicke identified that damage to the left temporal lobe impaired language comprehension while preserving speech production, establishing Wernicke's area as critical for language understanding (Wernicke, 1874).

In parallel, the systematic application of electric stimulations across the cerebral cortex provided crucial experimental validation in non-human experiments. Fritsch and Hitzig (1870)

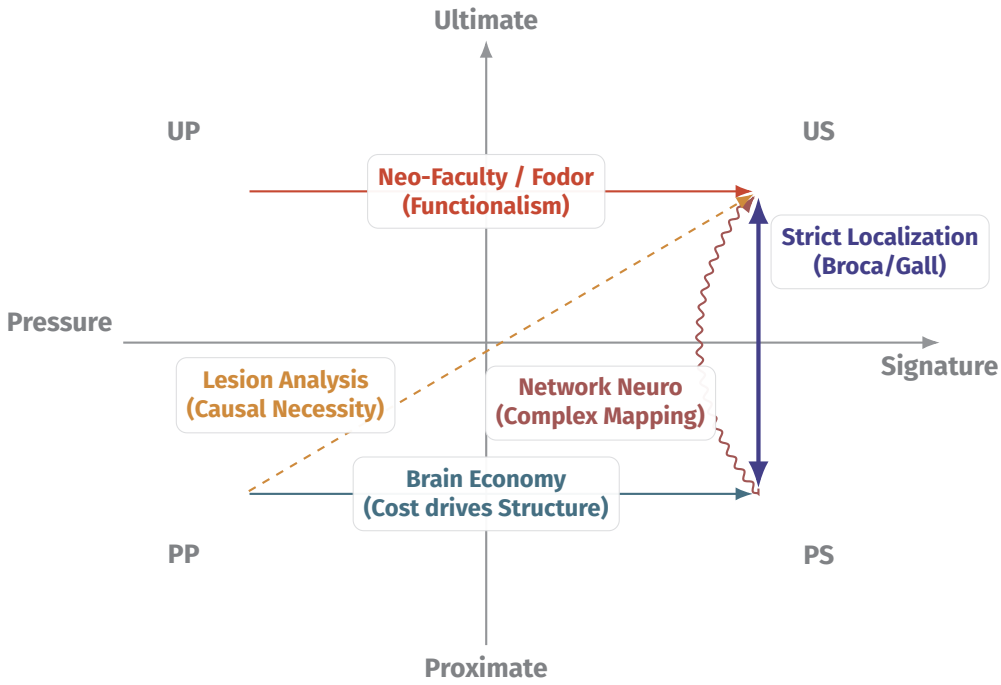


Figure 0.1: A small history of neuroscience research

were thus able in 1870 to successfully apply this method and identified a small region in the dogs' cortices that, when stimulated, produced small but reproducible twitches on the opposing side of the dogs' bodies. This was further confirmed by lesion analysis, confirming the functional importance of the newly found area: ablating sites that had evoked a certain movement type did impair the ability of subjects to generate said movements. This was then replicated in macaque studies, identifying similar regions in the cortex of monkeys (Beevor and Horsley, 1887; Ferrier, 1875).

Brodmann later came along and, investigating cytoarchitecture, mapped the cortex into distinct areas (creating a famous and extremely long-lived cortical map, or so-called Brodmann's areas). This cemented the idea that the brain contained *organs*, and that these organs could be found *structurally* and should display clear *functional* roles (Brodmann, 1909). It is to be noted that Brodmann himself was fairly nuanced in his views on the strict localization of function, but that in time much of the field has nonetheless adopted a fairly rigid modular mapping in his wake (Zilles, 2018).

0.5.2 *The Holist Counter-Movement* (PS $\neq$ US)

A concurrent school of thought, "holism" or "connectionism," challenged this fragmentation. Researchers like Karl Lashley proposed the principle of *equipotentiality* Lashley (1930), suggesting that areas of the cortex were functionally interchangeable, and able to compensate for

one-another if necessary. In this framework, while the brain possesses a modular structure (**PS**), its functional signature (**US**) is not a property of local modules but of the aggregate mass, making it arguably fall-out of any modular representation. This view effectively argued for a decoupling of local structure and specific function, positing instead a much more distributed organization.

0.5.3 *The Neo-Faculty School (UP → US)*

With time came the rise of a new doctrine, seemingly separating the study of the structural to the functional. This perspective could be seen as the most direct descendants of the aforementioned pioneers, yet somewhat abstracted away from the biological substrate. Their core focus has instead been on the cognitive architecture (**US**) required to solve the adaptive problems of our environment (**UP**). The cognitive revolution of the 1960s-80s revitalized the framework of Faculty Psychology, with Noam Chomsky arguably serving as the father of the modern revival. Bechtel and Richardson (1993) rightfully note that Chomsky's proposal of a "mental organ" for language was strikingly reminiscent of faculty psychology. Indeed, Chomsky postulated that the mind must possess innate, specialized structures to handle specific tasks (specifically language), effectively reviving the modular tone of the 19th century without its phrenological baggage. This fundamentally differed from previous influential thinkers such as B.F Skinner or Jean Piaget, describing the mind as a homogeneous undifferentiated "blank slate" or "black box", with little to no internal structures (Chomsky, 1980). Chomsky's insistence on the innate nature of such structures is important here, and the notion will be revisited later on.

Jerry Fodor further codified this notion in *The Modularity of Mind*. He proposed that the mind is indeed not a unitary learning machine but is composed, in part, of domain-specific, encapsulated modules acting as autonomous "vertical faculties." Such modules are characterized as fast, hard-wired, highly specialized, and mandatory: operating independently of one's conscious beliefs (explaining phenomena such as persistent visual illusions) (Fodor, 1983). Critically, Fodor argued (both early-on, and in response to latter extensions of the theory) that only the input systems of the brain fitted this description, while the "central" components remained non-modular (Fodor, 2000) . It is also worth noting that Fodor, at least in the early stages of his theory, maintained a belief in the strict neural localization of these modules, making him less of a 'functionalist' than some thinkers described in the rest of this section. He nonetheless later narrowed down the two essential features of modules to *domain-specificity* and *information encapsulation*, slowly abstracting away from substrate considerations, like the rest of the field.

Modern Evolutionary Psychology later extended this logic to the entirety of cognitive processes, arguing that the modular nature of the mind is a direct consequence of natural selection, a notion sometimes dubbed Massive Modularity (Barrett and Kurzban, 2006). Just as

the body evolved distinct organs for digestion and circulation, the mind evolved specialized computational tools to handle distinct selection pressures. The emphasis was now less made of the strict *Fodorian* definition of modules, and more centrally on *functional specialization* as the key principle under which to investigate these (Frankenhuis and Ploeger, 2007).

0.5.4 *A Modern but Partial Synthesis: Network Neuroscience (*PS $\overset{?}{\leftrightarrow}$ US*)*

Modern neuroscience has continued to attempt to link the modular signatures of the physical connectome to its functional dynamics, yet has adopted a much more nuanced view of this relationship. Having moved beyond the historical schism of localized versus distributed processing, neuroscience now understands this coupling as a complex, dynamical, and hierarchical one across multiple scales. The integration of graph theory tools into what is now referred to as "Network Neuroscience" has enabled a significant refinement of both the structural and functional signatures investigated in the brain (Bassett and Sporns, 2017; Sporns, 2018; Yu et al., 2018).

At the structural level, there is now strong agreement that the topology of the brain is well characterized by a small-world (Watts and Strogatz, 1998), structurally modular organization, while still displaying critical elements such as "rich clubs" and long-range connections (Sporns, 2011; Sporns and Betzel, 2016; van den Heuvel et al., 2012). At the 'functional' level, in part due to the advent of fMRI, systematic co-activation among regions has been the principal focus of study for researchers trying to understand functional dynamics, under the term Functional Connectivity (FC). In this sector, an emphasis has been placed on hierarchy, as well as on the dynamical and context-dependent nature of discovered 'functional networks', yielding insights into the dynamic reconfiguration of brain networks over time (Meunier et al., 2009; Sporns and Betzel, 2016; Thomas Yeo et al., 2011).

Critically, it is now widely admitted that structural and functional modules do not show as strong a coupling as previously assumed, with some areas displaying strong functional correlation even without direct physical connections. While structure *constrains* function, it does not dictate a one-to-one mapping (Damoiseaux and Greicius, 2009). Specifically, while activity is well predicted by structure during the resting state, this mapping quickly deteriorates when individuals are engaged in tasks. This suggests that fluid cognition requires more than a static, rigid structure to be understood. The structural connectome might act as a scaffold for function, but the brain is able to functionally "break away" from this rigid anatomy through local heterogeneity, higher-order interactions, and neuromodulation (Suárez et al., 2020). In order to quantify this more nuanced relationship, functional connectivity has been explored in terms of "gradients", with spatial gradients being discovered as a good predictor of function (Zhang et al., 2019). It has even been suggested that the strength of the coupling between SC and FC itself could be shown to follow a macroscale gradient, with sensory regions being more

strongly coupled than higher-level cognitive ones Preti and Van De Ville (2019). Understanding higher-order interactions of structure to function thus remains to be properly investigated.

0.5.5 *The Conflation of Property and Purpose*

Despite the advances in modern neuroscience, a definitional problem remains. In the majority of this literature, "function" is defined as *Functional Connectivity* (FC): the statistical correlation between activity in different regions. Suárez et al. (2020) argue for a crucial distinction between structure-function and structure-*property* relationships. They note that in complex systems, a property is a measurable feature (like the mass of a protein), whereas a function is a purposeful interaction (like the protein binding to a molecule). By this standard, resting-state FC is likely not the "function" of the brain, but rather a *manifestation* of the underlying process: a "hum" of the machine rather than the "work" it performs.

In the context of our framework, FC sits in an uncomfortable middle ground. It is often treated as a functional signature, yet it is merely a statistical readout of activity, devoid of the *intentionality* or *outcome* associated with biological survival. This limitation renders observational neuroscience somewhat "narrow" in its study of the structure-function relationship. Moreover, such an observational approach lacks the causal implications one would expect from a wider encompassing theory, constraining it to the right side of Table 0.1.

0.5.6 *Retaining Causality: Lesion Analysis* ($\mathsf{PP} \to \mathsf{US}$)

If functional connectivity provides only a correlational "hum," how to recover the causal link to true function? Historically this was the domain of lesion studies. In the context of our framework, a lesion represents a shift from *observation* to *intervention*. When a researcher (or an accident) lesions a brain area, they are effectively imposing a severe pressure: a hard physical constraint dictating that the system must now operate *without* a specific part of its substrate. Unlike the correlational approach of Network Neuroscience (where we observe the structure and the function simultaneously), lesion analysis manipulates the input (**PP**) to observe the resulting deficit in the functional output (**US**).

If region X is damaged and faculty A vanishes, but faculty B remains, then region X might be causally necessary for A. This is called a *single dissociation*. Conversely, if another link is observed, where region Y is damaged and this in turn affects B but not A, we have a *double dissociation*. For a large part of the field, this has been a clear indicator of functional specialization. Nevertheless, there is a reason this has slowly given way to the advent of fMRI research: First, lesions from naturally occurring injuries (a requirement for human studies) are hard to come by. More importantly, the very methodology of single-element perturbation

studies has been questioned, even when studying double dissociations, casting doubt on the real causal implications of such attempts (Fakhar and Hilgetag, 2022; Jonas and Kording, 2017; Young et al., 2000).

0.5.7 *The Brain Economy Principle* (PP → PS)

Conversely, while the modular structure of the brain has long been observed and thoroughly investigated, few have proposed further explanations as to *why* it is that way. An evolutionary argument could go along the lines of: Modular structure (**PS**) is a good support for Modular Functions (**US**), which in turn are needed to evolve in varying compositional environments (**UP**); hence, evolution has favoured modular brain structures. While potentially valid (although as we'll see some links in this explanation still remain unclear), an opposing, and somewhat more pragmatic school of thought has been developed.

The brain economy principle posits that the brain's topology and connectome are principally driven by physical limits. Proximate Pressures (**PP**), specifically the high metabolic cost of long-distance connections, are in this case placed as the primary driver of the brain's Proximate Signature (**PS**), characterized by high local clustering and small-worldness (Bassett et al., 2010; Bullmore and Sporns, 2012*a*; Horvát et al., 2016; Roberts et al., 2016; Samu et al., 2014). This mechanistic explanation is both compact and far-reaching, understanding the complex structures seen in biological neural networks from first principles, in a true *emergent* fashion. Any subsequent *functional* advantages displayed by such a modular structure would only be a by-product of some prior lower-level objective. However, this optimization (of both placement and connections) in the brain empirically seems to be only *quasi-optimal*, suggesting a trade-off between physical constraints and other (possibly functional / computational) demands (Kaiser and Hilgetag, 2004, 2006). This has also opened up the avenue for *generative* models of the connectome, often relying on such spatial embeddings, penalties and connection rules (Betzel et al., 2016; Vértes et al., 2012), an avenue that we'll explore further down the line section 0.7.2.

0.5.8 *Everything, Everywhere, all at Once ?*

Echoing the previous clash between localized and distributed, debates still exist regarding the proper emerging signatures of the brain. Coming full circle to the holistic counter-movement (section 0.5.2), and against the grain of a large portion of the field, recent theories have contested the very notion of a "functional module" and of modularity as a lens through which to study the brain as a whole. Instead, it has been suggested that we need to view the brain as a fundamentally complex and "entangled" system (Pessoa, 2022).

Neural Reuse theories (Anderson, 2010) and Neuronal Recycling (Dehaene and Cohen, 2007) suggest that our brain is not a static "Swiss Army Knife" of dedicated modules, but rather a dynamic "Lego set". In this view, local neural circuits are not specialized for complex tasks but rather constitute a set of (potentially innate) reusable "computational building blocks", displaying a diversity of neural patterns and distinct responses (Dehaene, 2020). These can be dynamically assembled and re-assembled across different networks to support various higher-level behaviours; consequently, what matters is not the areas involved (nodes) but the precise pattern of connections (edges) of the resulting computational graphs (Anderson, 2014). Such local circuits (the nodes) would not display direct specialization, and as such, it could be more accurate to speak of *functional palettes* or *profiles* (Anderson et al., 2013). In such a case, older areas (in the evolutionary sense) would get repurposed more frequently, leading to a more diverse functional repertoire. This idea directly contradicts the previous thinking that older areas, such as the visual cortex, are amongst the most specialized, yet it finds good experimental support (Anderson, 2010). In the context of modularity, this aligns with a definition of the concept not as *isolation*, but as *composability*, a distinction made by Bechtel and Richardson (1993) under the name of *simple* vs *complex* localization, while admittedly falling further away from the general umbrella of modularity as a concept.

Going even further, modern large-scale recordings in rodents have suggested that even minor, precise motor behaviours could be shown to have a remarkably large and decentralized impact on many areas of the subjects' brains (Musall, Kaufman, Juavinett, Gluf and Churchland, 2019; Stringer et al., 2019). If this were to generalize, most variables of use to animals could be shown to be encoded almost everywhere, making discussions about the functional roles of specific areas miss the point (Rosen and Freedman, 2025). In light of this, it has been suggested that neuroscience must move away from cortical maps altogether (Hayden, 2022). As such, a strong push of a part of field, renewing with a certain holistic tradition (although in spirit more than in practice) tends to emphasise the need of a new antimodularist and antilocalizationist narrative, with a "more flexible framework rather than a single organizational principle" (Hayden et al., 2025; Noble et al., 2024; Pessoa et al., 2021)

0.6 THE NEURO-AI OPPORTUNITY: FROM OBSERVATION TO CAUSATION

A fundamental limitation of the biological sciences is that they are primarily *observational*. We observe the brain at the end of a long evolutionary trajectory, forcing us to infer its causal history. As we have seen in the case of *lesion analysis*, the data available to recover causal links can be limited and rigid, while other proposed causal explanations (e.g., for the emergence of modular structure) often remain theoretical, lacking models to empirically test them. Artificial Neural Networks (ANNs), as well as other computational models, offer a unique opportunity

to invert this relationship. In such cases, as in the rest of this thesis, we can step into the role of the "causal actor."

0.6.1 *Mapping Experiments, Not Just Concepts*

The strength of the modularity framework described in this thesis is that it does not merely classify abstract concepts, but rather **experimental designs**. It creates an "Experimental Snapshot" of the logic in any given study. For example:

- In one experiment, structural modularity (as measured by the graph-based Q metric Newman, 2006) may be fixed as a **Causal Pressure** (e.g., a structural binary mask imposed on the neural network's connection, that's for example the case in Chapter 1).
- In another experiment, the same metric of modularity might be measured as an **Emergent Signature (PS)** of a training process (that's for example the case in Chapter 2).

By explicitly defining which quadrants are "frozen" as inputs and which are "free" as outputs, we can resolve apparent contradictions in the literature, all the while sanitizing the debate and making cleaner claims.

0.6.2 *The Experimenter as "Evolution Short-Circuiter"*

Over evolutionary timescales, this framework becomes a dynamic loop. A property that begins as an "Emergent Signature" (**US**) in one generation—a learned solution—can, if consistently beneficial, create a selective pressure. This could relate to the Baldwin Effect (Simpson, 1953), where beneficial *learned* signatures then create selective pressure for *innate* traits in future generations.

While biological evolution must optimize this entire causal chain ($\mathsf{PP} + \mathsf{UP} \xrightarrow{?} \mathsf{PS} \xrightarrow{?} \mathsf{US}$) simultaneously, creating a noisy and entangled system, we as experimenters can **short-circuit** this loop. We in contrast can isolate specific causal links. For instance, we can effectively "hard-code" the results of millions of years of structural evolution by imposing a modular prior (**PP**), allowing us to test its specific functional impact (**US**) in isolation. We act as the agents of a rapid, artificial Baldwin Effect, baking learned outputs into architectural inputs. This allows us to deconstruct the complex phenomenon of emergence into testable, independent interactions.

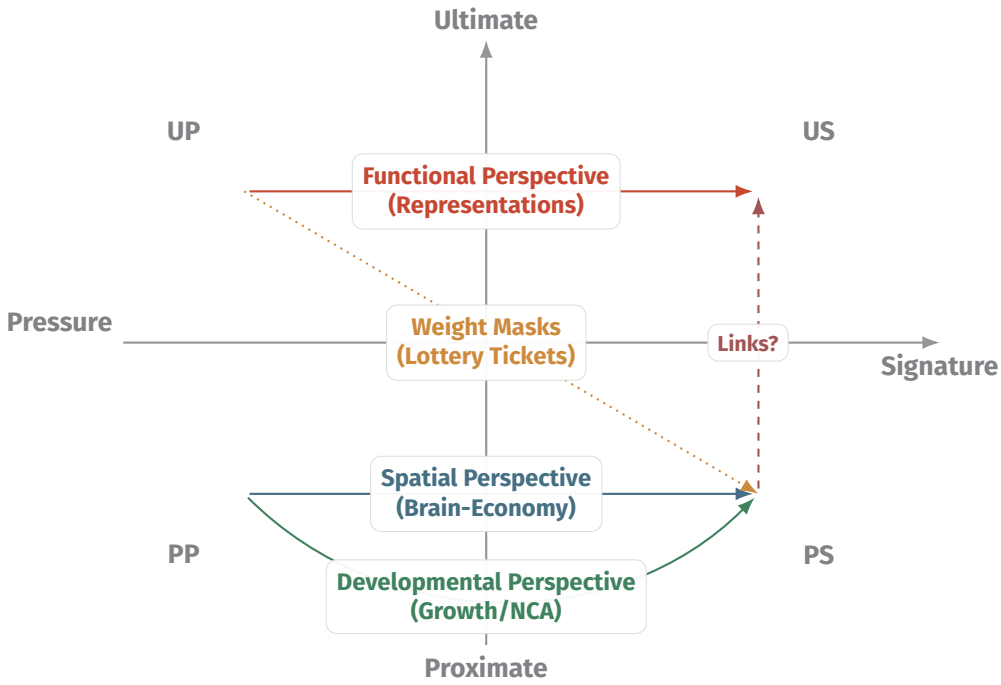


Figure 0.2: Studies and usages of emerging modularity in AI research.

0.7 STATE OF THE ART: THE COMPUTATIONAL EXPLORATION

A growing part of the Machine Learning community has begun to explore links within the quadrants defined above, effectively running "simulations" of biological forces to observe what architectures arise. However, the motivation for using modularity in computational systems is not uniform. Broadly speaking, we can categorize these efforts into two distinct routes based on their causal approach: those who *impose* modularity to solve complex tasks, and those who seek to *induce* it to understand its origins (while still potentially reaping its rewards down the line).

0.7.1 *Route 1: The Architects*

The first route views modularity as a robust and beneficial design choice, particularly when requiring systems that display adaptability, compositionality, and scaling capabilities (see section 0.1). Consequently, a vast body of literature has adopted modularity as a guiding engineering principle. This includes research in continual learning, transfer learning, meta-learning, and generalization, where the goal is often to prevent interference between tasks or to facilitate the reuse of learned knowledge. As such, the literature is vast, and the approaches designed to tackle such problems numerous.

While many architectures could be described as *implicitly* modular, a subset of the Deep Learning field has recently pushed for modularity as an *explicit* architectural prior, theorizing it as a necessity for the next generation of general intelligence (Amer and Maul, 2019; Ha and Tang, 2022; Pfeiffer et al., 2024; Salatiello, 2025). In this context, modularity is not an emergent outcome but a set of deliberate structural decisions. In particular, recent reviews have provided comprehensive taxonomies of this landscape, defining the critical design axes available to the neural architect (Amer and Maul, 2019; Pfeiffer et al., 2024). Drawing from these frameworks, we can identify five key dimensions where explicit choices must be made to construct a modular system:

- **Problem Decomposition (Domain):** Before a network is even constructed, a choice is often made regarding the modularity of the problem itself. Amer and Maul (2019) rightfully suggest that while most modular networks assume a modular environment, theorizing how such domain is actually created and segmented is crucial. These choices are relevant for *emergent* approaches as well, wishing to study systems tacking modular tasks and environments.
- **Module Definition (Computation):** How are the fundamental units of computation implemented? Definitions range from completely independent neural networks (ensembles) to using specific sub-layers, or even modulation of blocks within a larger transformer, by manipulating the inputs each module receives (prompt injection, input concatenation).
- **Routing Mechanisms:** Perhaps the most critical component of explicit modularity is the selection mechanism: how is the correct module chosen for a given input? Routing has been conceptualized as a general framework for creating flexible architectures, where a controller determines the path of data through the network (Rosenbaum et al., 2017). This routing can be hard-coded or learned dynamically along with the modules. Learning routing presents the fundamental difficulty of avoiding *module collapse* and instabilities during training. In that sense, utilizing the entirety of a module pool efficiently and adequately is a hard problem to learn (Rosenbaum et al., 2019).
- **Aggregation (or Integration):** Once modules have processed information, how are their outputs combined? Strategies range from simple weighted averages, to voting mechanisms and complex, learned non-linear combinations that fuse module outputs into a final prediction (Amer and Maul, 2019). Again, both fixed and learnt approaches exist.
- **Training Dynamics (or Formation):** Finally, how is the modular system optimized? Design choices here include whether to train modules and routers jointly (end-to-end), whether to freeze certain modules to protect them from catastrophic forgetting, or how to introduce new modules dynamically during the training process.

Under this taxonomy, Pfeiffer et al. (2024) have been able to study numerous approaches in a unified way. By navigating these axes, "Architects" have successfully engineered complexity

by building it up from more manageable, independent sub-systems. However, while this approach yields high-performing systems, it often bypasses the question of *origin*. In these systems, modularity is a premise, not a conclusion. This thesis, in contrast, aligns less with the engineering goal of optimizing performance, and more with the scientific goal of explanation. We do not ask how to *build* a modular machine, but rather: what pressures a system to *become* modular?

0.7.2 *Route 2: The Emergentists*

While architects carefully design modular architectures with the goal of creating efficient and adaptable learning systems, a secondary approach, perhaps more biased towards understanding than performing, attempts to study the *emergence* of modularity from first principles. This "Emergentist" perspective treats the neural network not as a product to be engineered, but as a substrate to be observed, looking for the conditions under which modularity arises "naturally".

Weight Analysis, Sub-masks, and the "Lottery Ticket" hypothesis ($\mathsf{UP} \rightarrow \mathsf{PS} \overset{?}{\leftrightarrow} \mathsf{US}$)

The most straightforward way to analyse the emergence of modularity is to look for it directly in the units and weights of trained (or untrained) neural-networks. Studying neural-networks as abstract graphs is an effective way to apply metrics from graph theory, such as structural modularity Q or small-worldness, and makes intuitive sense when talking about modules (being clusters of neurons).

The "Lottery Ticket Hypothesis" (Frankle and Carbin, 2019) made a case that interesting subsets of units exist in any initialization of an ANN. The hypothesis is that within a dense, randomly initialized neural network, there exists a smaller subnetwork (a "winning ticket") that, when trained in isolation, can match the accuracy of the original network. This observation shifted the analytical focus from global optimization to the identification of specific, sparse subsets of connections. Interestingly, the "potentiality" of these winning tickets have been shown to be quite robust and general. Tickets discovered to be highly performant (after training) on a certain task were shown to tansfer remarkably good performances across other tasks and datasets (still requiring re-training nonetheless, the transferred quantity being in that case the mask and initial random weights) (Mehta, 2019; Morcos et al., 2019; Soelen and Sheppard, 2019).

Malach et al. (2020); Ramanujan et al. (2020) subsequently proved an even stronger point: within a sufficiently wide, randomly initialized network, one can find a subnetwork that achieves performance comparable to the trained target network *without any further weight training*. Here, the "learning" process is reframed not as adjusting synaptic weights, but as finding the correct binary mask—effectively identifying a functional topology hidden within

random noise. Wortsman et al. (2020) leveraged this insight to design a novel continual learning framework. By learning a task-specific "supermask" (a differentiable weight mask) for each incoming task, a single large network can support a multitude of task variations with minimal interference. In this light, we begin to see a potential link between these tickets and modularity: unstructured, task-specific subsets of neurons (lottery tickets) could be interpreted as budding modules, and differentiable weight-mask could prove a fruitful way of identifying them.

However, do these sparse topological subsets constitute true modularity? Csordás et al. (2021) investigated this by applying weight masks to networks trained on multiple tasks to probe their emergent modular nature. They discovered that while "solution-implementing modules" (specialized subsets of weights) do emerge, they fail to reflect the general architecture of the tasks. Critically, networks by default lack the ability to *reuse* modules appropriately across tasks. In this case, a failure to share and recombine subcomponents prevents the computational efficiency and generalization capabilities expected from properly modular algorithms. Investigating larger networks on an odd-one-out compositional task, Lepori et al. (2023) claim to demonstrate a significant amount of "structural compositionality", by successfully discovering subnetworks implementing specific "subroutines" of a larger compositional task. They do not, however, investigate whether such subnetworks are effectively reused *across* tasks implementing the same subroutines, effectively missing the point raised by Csordás et al. (2021), while nonetheless re-affirming the existence of *specialized* subnetworks. Other algorithms for detecting "local specialization" in deep neural networks have also been developed, hinting at a similar conclusion (Hod et al., 2022). Thus, while sparse subnetworks in dense graphs offer a form of specialization, they fall short of a robust definition of modularity. It appears that finding "lottery tickets" alone is insufficient; additional biases are required to force these neuronal clusters to organize into robust, compositional structures.

The existence of lottery ticket is in a way reminiscent of an argument from Dehaene (2020): in their view we are natively endowed with a wide repertoire of diverse neuronal assemblies, and subsequent learning taps in that potential available at "initialization". Lottery tickets could be a blunt, unconstrained first approximation of such a concept, and could be substantially refined in a true meta-learning scenario, to enable architectures initialized with an ensemble of high-potential subnetworks, or modules, ready to be fine-tuned and recombined.

The Functional Perspective ($\mathsf{UP} \rightarrow \mathsf{US} \overset{?}{\leftrightarrow} \mathsf{PS}$*)*

A separate and significant body of work has focused in contrast on the emergence of *representations* in recurrent neural networks (RNNs) driven purely by task demands. When training unstructured RNNs on multiple compositional tasks, network dynamics have been shown to self-organize into abstract, factorized representations in high-dimensional activity space (Driscoll et al., 2024; Johnston and Fusi, 2023; Sandbrink et al., 2024; Yang, 2019). In this context, the word "representations" is crucial, marking a shift to a "higher-level" analysis than merely

studying assemblies or structures of neurons. From this perspective, the relevant signatures of interest lie in the complex dynamics of units, and critically in the dimensionality reduction of a very high-dimensional signal (one dimension per unit) into lower-dimensional manifolds potentially informative of the tasks at hand.

Driscoll et al. (2024) called these "computational motifs", and describe them as a crucial unit of computation, "intermediate between neuron and network." In such cases, whether these motifs (or alternatively precise subspaces of activity, dubbed by Johnston and Fusi (2024) as *implicit* modularity) are supported by distinct clusters of units (conversely termed *explicit* modularity) remains to be investigated. However, the key shift from substrate space to activity space is to be noted. As is common with higher-level emergent objects, computational motifs possess properties that their low-level support does not, making them worthy of investigation: the geometry and dynamics of these multidimensional structures can provide a new kind of mechanistic insight into the inner workings of neural computation.

Going beyond multitask pressure, networks have also been shown to develop modular structures when pushed to predict future inputs in a predictive coding scheme (Bakhtiari et al., 2021; Boeshertz and Clopath, 2025). This highlights the importance of pinpointing the correct Ultimate Pressures required for modularity to 'naturally' emerge. Similarly, varying environments (or "modularly varying goals", MVG) have been theorized as a necessary condition for this emergence (He et al., 2009; Kashtan et al., 2007). However, while high-dimensional representations in these studies display a form of functional modularity, the findings are largely substrate-independent (often by choice, as Johnston and Fusi (2024) purposefully investigate whether computational constraints *alone* can drive modularity). They focus on the "software" while often ignoring the constraints of the physical "hardware", and affirming that specific identity of the nodes is secondary to the manifold they collectively generate.

The Spatially Embedded Perspective ($\mathsf{PP}_{\mathtt{spatial}} \rightarrow \mathsf{PS} \overset{?}{\leftrightarrow} \mathsf{US}$)

In contrast to such a substrate-independent approach, a second stream of research re-introduces physical constraints, effectively simulating the "Brain Economy" (section 0.5.7) hypothesis in silico. The idea that short-range connections could confer a beneficial computational advantage is not new (Jacobs and Jordan, 1992). However, it was revitalized by a series of papers from Clune et al. (2013), showing that a simple connection cost penalty could lead to the emergence of modular and even hierarchical structures.

Crucially, Ellefsen et al. (2015); Mengistu et al. (2016) demonstrated that this structurally modular organization, born from *proximate* pressure, yielded beneficial *ultimate* advantages in terms of adaptability and performance. This serves as a remarkable demonstration of the causal chain: by applying a Proximate Pressure (**PP**), they observed a Proximate Signature (**PS**), which facilitated an Ultimate Signature (**US**). Notably, these networks were not directly

pressured to be adaptable during the selection process; rather, adaptability emerged as a "free" by-product of the structural constraint. This very process was further demonstrated using the HyperNEAT algorithm (Stanley et al., 2009), known to produce structures with regularities, combined with the connection cost technique, producing "regular, modular and hierarchical" structures (Huizinga et al., 2014, 2016).

This line of thinking, that spatial sparsity can be utilized to gain functional advantages and more interpretable networks, has since been explored and extended in several recent works. Achterberg et al. (2023) recently revitalized the concept of what they dub "spatially embedded RNNs", effectively training recurrent networks with an added distance-based regularization term. They showed that this in turn led to modular small-world topologies, in line with previous work. In addition, they observed interesting functional signatures echoing neuroscience findings, such as mixed-selective codes, and detected alignment between functional and structural modules. Interestingly, they required an additional regularization term promoting good graph connectivity alongside the connection costs to achieve high-performing results, suggesting that only penalizing long-distance connections might be too blunt of an approach.

Going beyond fixed neuron positions, Mészáros et al. (2025) critically reframes the debate by reminding us that space does not only constrain networks through energy costs; it also endows them with heterogeneous delays, a vital property thought to have strong functional implications (Sun et al., 2025). Learning neuron positions while minimizing a distance-based connection cost would inevitably lead to a "collapse" of neurons into a single position, were it not for an "opposing force" of sorts. Mészáros et al. (2025) show that delays can act as such a force, and that networks trained on temporal classification tasks naturally organize into clustered structures, balancing the costs of segregating modules with the advantage of having delayed computation between them, hinting again at the fundamental *trade-off* described in section 0.5.7.

In a more "performance-driven" approach, and re-introducing ideas from the lottery-ticket hypothesis, it has been demonstrated that the addition of a distance penalty improved the overall performance of sparse networks, both when pruning them or when using rewiring techniques to keep sparsity constant (Keirsbilck et al., 2019; Khona et al., 2023; Zhang et al., 2025). In such cases, connection cost appears to be a simple yet effective prior to add to sparse neural network training. Beyond performance, the field of *mechanistic interpretability* has also investigated this idea, showing that spatially embedded networks lead to more visually interpretable solutions (Liu et al., 2023). It remains to be investigated whether such sparse subnetworks, which now naturally align with spatial clustering, display the hallmarks of modular organization that were shown to be absent in standard lottery tickets networks.

The Developmental Perspective ($\mathsf{PP}_{\mathtt{growth}} \rightarrow \mathsf{PS} \overset{?}{\leftrightarrow} \mathsf{US}$)

Finally, there exists a crucial, yet potentially under-studied low-level prior: the developmental process itself. Biological neural networks do not appear *ex nihilo* as tabula rasa structures; they must be *grown*, and possess *innate* structures. This introduces two concepts: the *genomic bottleneck* and *algorithmic growth*. While not usually linked in the literature, these could reveal to be deeply connected.

First, the information available to guide the initialization of the brain is orders of magnitude smaller than the total information contained in the final synaptic connectome (Zador, 2019). Moreover, in the realm of biological brains, all are endowed with innate abilities; even if the extent of such abilities is debated in humans, there is no doubt that we do not escape this pattern. If we accept the hypothesis of innate structures (as discussed in section 0.5.3), reinforced by the genomic bottleneck idea, the differentiation of neurons into specialized substructures must originate at the genomic level. This modular mapping between genotype and phenotype is hypothesized to be a key enabler of *evolvability*, allowing systems to adapt without suffering from catastrophic interference, as discussed in section 0.1. Studying how neural networks can be endowed with innate abilities through different encodings has been a fruitful avenue of research (Koulakov et al., 2022; Stöckl et al., 2022), but has not clearly demonstrated a link with modularity. It could be the case that part of the story has been omitted: the genome does not only encode innate properties in an indirect encoding fashion; it is rather a developmental encoding: it also encodes the rules to *grow* the network.

The fact that biological structures need to be grown is both a fundamental property and limitation. This constraint could be deeply linked to the concept of *algorithmic growth* seen in Cellular Automata (CA). While the effective information encoded in the *rules* of development might be limited (just like the rules of a simple cellular automaton contain only a few bits), letting it unroll in time progressively leads to more and more complex *final structures*. This means that, while the *algorithmic information* — in the sense of Kolmogorov (1968); Solomonoff (1964) — of a CA state remains very low, the time and energy needed to unfold this information — or alternatively its "logical depth" as defined by Bennett (1995) — is high, resulting in what is dubbed a high *endpoint information* (Hiesinger, 2021). Moreover, encoding the recipe for a complex brain into a compressed genome might necessitate the creation of regularities, synthesis, and hierarchies that the developmental process can tap into. Furthermore, the fact that this development occurs in physical space inevitably engenders fractal, scale-free properties reminiscent of the small-world modularity observed in biological networks.

Recently, cellular automata have been rejuvenated through the concept of *Neural Cellular Automata (NCA)* (Mordvintsev et al., 2020): in such a setup, the local rule governing the evolution of the cell's state is a parametrized neural network. The process of applying this neural network (or *rule*) to all cells in a recurrent manner becomes a differentiable model of morphogenesis,

allowing us to learn said rules end-to-end to steer the computation into the desired final state. In the original paper, the endpoint is supervised (e.g., an RGB image), but biology fundamentally differs in this regard, for growing a neural network is really only the beginning. This new implicit, or *functionalist* perspective, as coined by Turing and evoked in Agüera y Arcas (2025), radically changes the inherent goal of the developmental problem, turning it into a *meta-learning* setup where the developmental system must attain good initialization to tackle subsequent problems. Najarro et al. (2022*a*, 2023) successfully applied NCA to the problem of growing functional networks in a method called Neural Developmental Programs (NDP), and were in fact capable of displaying "innate" performance. They were also able to grow small-world graphs when explicitly training the model to do so. It remains unknown, however, whether a NDP of this sort could incorporate spatial considerations to naturally grow modular structures, that in turn display strong innate potentiality for subsequent learning.

0.8 ROADMAP OF THE THESIS

Having established the general framework of modularity and disentangled the causal links investigated in the literature, we now situate the contributions of this thesis within this landscape.

Chapter 1, **"Dynamics of specialization in neural modules under resource constraints,"** confronts the fundamental challenge of defining and measuring functional modularity. It serves as a necessary calibration step. We explicitly impose structural modularity as a prior to investigate its direct effect on functional specialization (taking on the role of the "experimentalist as evolutionary short-circuiter described in section 0.6). This investigation reveals a surprising decoupling: structural boundaries do not guarantee functional separation. We find that the tight link usually assumed between structure and function really only holds under the application of specific resource constraints and environmental separability. We furthermore probe into the *dynamic* nature of this relationship, investigating the coupling when presented with noisy stochastics inputs. This demonstrates that the map between the physical and the functional is not rigid, but loose and dynamic.

This sets the stage for Chapter 2, **"Spatial And Neuromorphic Priors for Compositional Learning,"** which turns to the problem of compositionality. Having established that constraints are necessary for specialization, we investigate the final physical constraint: *Space*. We try and understand the interplay between the emergence of structure in sparse, spatially-embedded recurrent networks, and their capacity to tackle a compositional task and generalize. We ground these abstract principles in physical reality by implementing architectures based on a memristor-based neuromorphic chip. We demonstrate that real energy constraints naturally induce modular systems that outperform unstructured sparse architectures in both performance and sample efficiency.

The second half of the thesis marks a departure from the study of fixed architectures and explicitly imposed constraints. We dive a level deeper to investigate the theoretical foundations of self-organization itself. Before we can truly understand how modularity "grows" or how brains develop, we require a better understanding of how complex, intelligent behaviours can arise from local interactions.

Chapter 3, **"A Path to Universal Neural Cellular Automata,"** is an exploration of the computational limits of self-organizing substrates. While Neural Cellular Automata (NCA) have typically been treated as tools for pattern formation (morphogenesis), here we challenge them to become general-purpose computers. We explore whether continuous systems driven by local rules can master computational primitives, effectively trying to understand how we can compile algorithms to "interface" with a continuous computing medium. This is not yet a study of biological growth, but rather a search for the conditions under which a continuous, locally-connected substrate can sustain universal computation.

Finally, Chapter 4, **"Self-Organising Digital Circuits,"** brings these principles closer to the realization of intelligent agents. We introduce a system where the local rules (implemented via a NCA Transformer) do not perform the task directly, but rather act as an optimizer for a separate functional substrate (a boolean digital circuit). By replacing global backpropagation with local message passing, we achieve a system that is robust to damage and capable of reconfiguring itself in the face of hardware failure. This touches on a fundamental principle of biological intelligence: the ability to maintain function through adaptive resilience. If the boolean gates were neurons, and the optimization process was a learning rule, this architecture could represent a brain that "knows how to learn," offering a glimpse into scalable, robust intelligence built entirely on local rules.

Together, these investigations reveal modularity not as a single property but as an emergent phenomenon arising from the tension between constraints and functions, between specialization and integration, between local rules and global behaviour. By controlling and understanding these multiple facets, we move closer to understanding how intelligence emerges from the interaction of simple parts.

Part I

INTELLIGENCE UNDER CONSTRAINTS: RESOURCES, SPACE, AND MODULARITY

1

DYNAMICS OF SPECIALIZATION IN NEURAL MODULES UNDER RESOURCE CONSTRAINTS

Narrative Preamble

In the taxonomy of modularity introduced in the Introduction, we pick up on a general conflation in the literature: the tendency to view structural modularity (clustering of the physical substrate) and functional specialization (decomposition of the task) as synonymous, or at least inextricably linked. While we've seen that there is significant pushback against such a rigid, one-to-one mapping, a fairly *localized* view still dominates how we think about the structure-function relationship of neural networks. Before we can engineer emergent systems, we must first rigorously test this assumption.

This chapter acts as a calibration of our framework. We occupy the role of the "Evolution Short-Circuiter" defined in section 0.6. By manually imposing structural modularity as an experimental input, we isolate its causal impact on the emergence of functional specialization. We ask a simple but probing question: Would a modular brain encourage a modular mind? By systematically varying environmental pressures and resource constraints, we demonstrate that structure is an insufficient condition for functional specialization, and we effectively map the transition zone where specialization does emerges.

Publishing Context

This paper arised from the first project of the PhD, back in 2021, and led to an early first pre-print : **Gabriel Béna** and Goodman (2021). We then picked up the project again in 2023, which after continual progress and substantial changes, became the final form of the paper, published in Nature Communications: **Gabriel Béna** and Goodman (2025).

1.1 ABSTRACT

The brain is structurally and functionally modular, although recent evidence has raised questions about the extent of both types of modularity. Using a simple, toy artificial neural network setup that allows for precise control, we find that structural modularity does not in general guarantee functional specialization (across multiple measures of specialization). Further, in this setup (1) specialization only emerges when features of the environment are meaningfully separable, (2) specialization preferentially emerges when the network is strongly resource-constrained, and (3) these findings are qualitatively similar across several different variations of network architectures. Finally, we show that functional specialization varies dynamically across time, and these dynamics depend on both the timing and bandwidth of information flow in the network. We conclude that a static notion of specialization is likely too simple a framework for understanding intelligence in situations of real-world complexity, from biology to brain-inspired neuromorphic systems.

1.2 INTRODUCTION

Modularity in the brain

Modularity is an enticing concept that naturally fits the way we attempt to understand and engineer complex systems. We tend to break down difficult concepts into smaller, more manageable parts. Modularity has a clear effect in terms of robustness and interpretability in such systems. Disentangled functionality means that the impairment of a module doesn't lead to the impairment of the whole, while making it easy to spot critical failure points. When conjoined with redundancy, modularity thus forms the basis of an extremely robust general organizational principle (Chen et al., 2021). Following the early observations of Brodmann (Brodmann, 1909) on cytoarchitecture, it has been thought to be an essential element in the organization of the brain: brain organs can be found structurally and should have distinct functional roles. Brodmann himself was no naive localizationist when it came to the brain, admitting that only very elementary functions could be completely localized, and that complex brain functions would only emerge through the interaction of many of those sub-organs (Zilles, 2018). Nevertheless, a fairly modular view of the brain has since prevailed. However, the precise link between structure and function has never been conclusively established, and recent work has begun to challenge this organizational principle (Bassett and Sporns, 2017; Pessoa, 2022).

As a more emergent principle linking structure and function, it has been suggested that modularity emerged as a byproduct of the evolutionary pressure to reduce the metabolic costs of building, maintaining and operating neurons and synapses. It has been shown that brain

networks are near-optimal when it comes to minimizing both its wiring and running metabolic costs. For example, the summed length of the wiring in the brain has been minimized (Raj and Chen, 2011), with nodes being near-optimally placed (Achard and Bullmore, 2007; Cherniak, 2012; Chklovskii et al., 2002; Striedter, 2005). At the same time, brains also display complex and efficient topology allowing for incredible information processing capabilities. To do so brain networks possess some very costly elements to maintain such as long-range connections, a 'rich-club' of densely connected neurons, and some non-optimal placement of components (Chen et al., 2013; Chklovskii, 2004; Heuvel and Sporns, 2011; Kaiser and Hilgetag, 2006). Based on these observations, brain organization could be understood as a trade-off between mitigating these inherent metabolic costs (from being a spatially embedded network) and its efficiency and capabilities of information processing (needed to solve complex tasks and display the emergent intelligence that we witness) (Bullmore and Sporns, 2012*a*).

Defining modularity

We thus have to distinguish two types of modularity, *structural* and *functional*, and understand how they are related. We take structural modularity to mean the degree to which a neural network is organized into discrete and differentiated modules. Such modules show much denser connections internally than across other modules. This is usually computed using the Q-metric (Newman, 2006), measuring how much more clustered these modules are when compared to a graph connected at random. Although note that many other techniques are possible, and that module detection in networks (Betzel, 2020*a*) as well as defining measures of modularity (Casper et al., 2022; Watanabe, 2018) are complex and interesting fields in their own right. While this structural definition is important, it doesn't necessarily inform us on the function of the modules.

Functional modularity refers to the degree to which potential modules – which may or may not be the same as structural modules – perform specialized and distinct functions. Potential definitions have been put forward, to try to conceptualize what should constitute modules. While the details have been contested, to frame our definitions it may be helpful to recall some of the properties of Fodor's definition (Fodor, 1983) refined in Shallice and Cooper (2011). This can help us pinpoint a few properties that a functionally modular system should possess, and allow us to formulate associated quantitative measures.

1. Modules should only carry a sub-function of a more complex overall function.
2. *Domain specificity* states that a module should respond to and operate only on specific types of inputs.
3. *Separate modifiability* means that the impairment of one module should not affect the functioning of another.

4. *Information encapsulation* asserts that a module should have restricted access to information outside its own state.

An important property, hinted at by these definitions, that we do not address directly in this study is the ability of a network to re-compose previously acquired (and encapsulated) knowledge to achieve systematic generalization. We touch on this important feature in the discussion.

In animal brains, functional modules are usually identified in two ways: by using data from neural activity or by studying the impact of lesions. The former requires large-scale (or even whole-brain) recordings, and consequently relies on examining co-activations of brain areas, based on a proxy measure. This would relate to point (2) - *domain specificity*. In the human brain, fMRI measures changes associated with blood-flow to infer neural-activity, that then serves as a basis to infer functional connectivity (Power et al., 2011; Sporns, 2013; Sporns and Betzel, 2016). In small vertebrates like zebrafish, that can take the form of calcium imaging (Betzel, 2020*b*; van der Plas et al., 2023; Vishwanathan et al., 2020). Nevertheless correlations can only tell us so much (as Ralph Adolphs and David Anderson would put it, imaging a car's speedometer to infer its speed wouldn't tell us much about the underlying mechanisms moving it forward (Adolphs and Anderson, 2018)).

Possibly the most authoritative way of examining the functional organization of the brain relies on lesion analysis. Studies examine naturally occurring injuries (in humans at least) to understand the effect that the impairment of an area has on behavior. This would relate to the concept of *separate modifiability*. Those lesions can however be hard to come by - even if it has been argued that a systematic collaboration across hospitals could provide a substantial lesion database (Adolphs, 2016). Moreover lesion analysis needs consistent cross-validation and control to correctly infer a functional link. It has been shown that single-lesion experiments could be insufficient to properly infer causation, or even be biased (Fakhar and Hilgetag, 2022; Jonas and Kording, 2016; Ke et al., 2021), and that this includes double dissociation studies relying on single-lesions injuries or manipulations (Young et al., 2000). Nevertheless, ablation-based analysis remains a central tool for understanding neural networks, both biological and artificial (Meyes et al., 2020).

Overall, gaining a rigorous causal understanding of brain functions seems to be a highly challenging problem. While all the aspects discussed above can help us conceptualize and measure specialization, they would also be challenging to clearly test in real intelligent systems. We've seen that an analysis of both activity and lesions could prove insufficient to provide a full functional understanding. This is where experiments on artificial neural networks (ANNs) could prove crucial, as both can be precisely and simultaneously applied (on the same networks), and where other aspects like points (1) and (4) can be investigated.

Our approach

A better understanding of the concepts of modularity *in silico* could result in fresh insights when it comes to natural brains for two reasons. First, ANNs have increasingly been proven to be valuable models of brain functions. Learning in both artificial and biological neural networks has to solve many of the same problems (like credit assignment: the problem of understanding how an individual element or parameter contributes to the performance of a network). Although the precise mechanisms used are likely to be different it has been suggested that the solutions could end up relatively similar, as biological brains could be approximating exact gradients in a local manner (Lillicrap et al., 2020; Whittington and Bogacz, 2019). Spiking neural networks (SNNs) have been developed, relying on a plausible way of communication between neurons, and implementing biological learning mechanisms, neuronal models, and general organization (Gerstner et al., 2014). ANNs have also proven remarkably able to predict (Yamins and DiCarlo, 2016) and decode (Musall, Urai, Sussillo and Churchland, 2019) patterns of neural activity in the brain. Finally, some strong patterns of similarity have been observed in the activity of biological and artificial neural networks (such as receptive fields or grid-cells neurons) (Banino et al., 2018; Whittington et al., 2018). This solidifies the idea that studying modularity in artificial networks could refine our understanding of it in the brain (as in, for example, Clune et al. (2013); Marton et al. (2021)).

Second, ANNs are fully and precisely controllable, and are in that sense the perfect testing ground to study structural and functional definitions of modularity. They can act as a sort of baseline, allowing us to discard measures and definitions that fail to give coherent results even in a perfectly controlled environment. Not only that, but ANNs increasingly incorporate modularity explicitly in their design, making the study of this concept especially relevant even in such an abstract setting. Recent deep learning models are modular in the sense that they are compositions of a few types of basic building blocks. However, it is not clear that they are functionally modular in the way the brain is thought to be (Csordás et al., 2021). Moreover, ANNs usually disregard the fact that biological networks are spatially embedded networks running under constraints, a feature we discussed earlier as a potential key to understanding neural network organization. Nevertheless, work has been put into developing a new general framework explicitly incorporating modularity into models, and the field is actively gaining momentum and attention (Ha and Tang, 2022; Pfeiffer et al., 2023).

To allow us to tease apart the factors influencing the relationship between structure and function, we designed flexible artificial neural networks performing tasks in a highly controlled synthetic environment. Generally, the link between structural and functional modularity is context-dependent and involves a complex and dynamic interplay of several internal and external variables. However, it is unclear the extent to which structural modularity is important for the emergence of specialization through training. We show here a case where even under

strict structural modularity conditions, modules exhibit entangled functional behaviors. We then explore the space of architectures, resource constraints, and environmental structure in a more systematic way, and find sets of necessary constraints (within our constrained setup) for the emergence of specialized functions in the modules. Finally, by measuring specialization in networks unrolled in time, we find that specialization actually has complex temporal dynamics governed by the dynamics of the inputs and communication between modules. Refining our conceptual understanding of modularity and specialization in this abstract setting may lead to fresh insights into the role of modularity in the brain and in brain-inspired algorithms and neuromorphic hardware.

1.3 METHODS

The goal of our work is to investigate the relationship between structural and functional modularity. To that end, we need to design flexible models, environments, and tasks, upon which we have precise control, to study this relationship in a controlled manner. We also need to understand and measure specialization and to do so we devise metrics that allow us to quantify this elusive concept.

1.3.1 *Environment: Data and Tasks*

Conceptually, we think of an environment as composed of underlying variables, some of which are crucial to the task at hand (which we will call *decision variables*), while the remainder are needed only to fully reconstruct the exact observation but can be discarded when performing the task (*irrelevant variables*). Put together, those variables fully specify the environment. However, in carrying out a task we are only interested in capturing some high-level features of the world: we therefore define global tasks to be computations on *decision variables*.

In this framework, recovering the *decision variables* from a stimulus is what constitutes the **sub-tasks**: the essential building blocks which can then be used to perform a more general **global-task**. The precise form of stimulus can vary, representing different sets of *irrelevant variables*. Changing the nature and structure of this stimulus is what allows us to control how difficult it is to recover a decision variable, hence the difficulty of the **sub-tasks**. We use (1) pairs of written MNIST (Deng, 2012) digits (with each digit drawn from the same set), or (2) pairs of EMNIST letters (Cohen et al., 2017) (with each letter drawn from separated subsets). The **global task** is the precise computation to be performed after decoding the **sub-tasks**, and can also vary, allowing for control over the amount of cooperation needed by the agents. We train the networks on a parity-based choice:

$$\mathcal{T}_\sigma = \mathcal{D}_1 \cdot \sigma + \mathcal{D}_2 \cdot (1-\sigma) \text{ where } \sigma = (\mathcal{D}_1 + \mathcal{D}_2)\%2 \tag{1.1}$$

$\mathcal{D}_1$ and $\mathcal{D}_2$ are the numerical values of both digits (or the indices of the letters in the case of E-MNIST), and σ is the parity of the sum, (ie $\sigma = 0$ when both are the same parity, and $\sigma = 1$ otherwise, with % being the modulo operator in Eq. 1). This task requires the network as a whole to classify both $\mathcal{D}_1$ and $\mathcal{D}_2$, but if done in a modular way each module can recognize only one digit and communicate only a single bit to the other: the parity of its own digit. This allows us to examine networks at extreme levels of sparse communication that can still solve the task. Let's take a look at the example in Figure 1.A to make the task clearer:

1. Two random digits $\mathcal{D}_1$ and $\mathcal{D}_2$ are sampled. In this case, $\mathcal{D}_1 = 0$ and $\mathcal{D}_2 = 3$.
2. Digits are fed to the network through their input weights (either all-to-all or one-to-one), at every time-step.
3. Modules then need to determine if digits are the same parity ($\sigma = 0$) or not ($\sigma = 1$). In this case, $\sigma = 1$.
4. In the scenario where $\sigma = 0$, the network must output the prediction $\mathcal{D}_1$. If not, the network must output $\mathcal{D}_2$. In this case, the correct label to this particular sample is $\mathcal{T}_\sigma = \mathcal{D}_2 = 3$.

Note that the task is unbalanced, since the case $\mathcal{D}_1 = \mathcal{D}_2$ provides information on both digits, making $\mathcal{D}_2 = \mathcal{T}_\sigma$ more frequent than $\mathcal{D}_1 = \mathcal{T}_\sigma$. To balance this, we train our networks to predict both:

$$\mathcal{T}_\sigma = \begin{cases} \mathcal{T}'_\sigma = \mathcal{D}_1 \cdot \sigma + \mathcal{D}_2 \cdot (1-\sigma) \\ \mathcal{T}''_\sigma = \mathcal{D}_1 \cdot (1-\sigma) + \mathcal{D}_2 \cdot \sigma \end{cases} \tag{1.2}$$

The models consist of recurrent neural networks to allow for communication between modules, and we run them for a number of time steps (between 2 and 5). In most cases, stimuli are presented identically at every time step, but in section 1.4.3 they are switched on at a random time step and then remain on (section A.5.2). In that section we also add independent noise to the inputs at each time step (see details in section A.5.1).

1.3.2 *Networks*

Architecture: We use a family of model architectures, composed of pairs of modules (single-layer recurrent neural networks of n neurons) that are densely connected internally and sparsely

connected between modules (a fixed fraction p of the possible synapses, leading to a number of synapses p_s). To be precise, these modules consist of vanilla RNNs, but the code is written to allow the use of other modules types, such as GRUs. The modules can either have separate or shared inputs, i.e. either each module only receives input from one digit image, or each module receives input from both digit images. Similarly, they can have either separate or shared outputs (pathway structure). See below for how the network decision is made in these two cases. We can either connect the module directly to its output or include a narrow bottleneck layer to restrict the bandwidth before readout. We vary n, p, the pathway structure, and the presence of the bottleneck layer. A typical network architecture is shown in fig. 1.1A, and all possible global architectures are shown in a minimal fashion in fig. 1.1B.

The choice of a recurrent rather than feed-forward architecture was made to keep a consistent architecture throughout the paper and to simplify the definitions of functional specialization and modularity.

Decision: Modules produce logits as their output, later used to compute the global network decision. As we've just seen, modules can either feed into a shared readout, (in which case the readout directly provides the network's decision), or have their own separate readouts. In the latter case, modules compete for decision via a winner-take-all scheme, to ensure that no communications is happening at the readout-layer level. In the separate readout case, each module is densely connected to a readout layer $\mathbf{r}^{(m)}$. Given the two readout layers, we compute the actual decision of the global model using a *max* decision, explained in the following paragraph:

$$\mathbf{r}^{out} = \mathbf{r}^{(\mu)} \text{ where } \mu = \arg\max_{m \in 0,1} \max \mathbf{r}^{(m)}. \tag{1.3}$$

For each input sample, the module with the highest overall "certainty" (largest value) takes the decision. More precisely, the global output layer of the network $\mathbf{r}^{out}$ is defined to be the output layer of the sub-network with the largest maximum value. In practice, that means that only the module having produced the largest overall value is being used in computing the resulting loss value, and that gradients are back-propagated to the other (non-deciding) module only via the (sparse) communication weights. This choice was made to minimize interactions between the modules other than the explicit sparse connectivity, ensuring the necessity of training the sparse inter-module weights.

1.3.3 *Structural modularity*

We define the fraction of connections between two modules of size n as $p \in [1/n^2, 1]$. The same fraction of connections is used in each direction. The smallest value is $p = 1/n^2$ corresponding

to a single connection in each direction, and the largest fraction $p = 1$ corresponds to n^2 connections in each direction (all-to-all connectivity).

We adapt the Q metric for structural modularity (Newman, 2006) to a directed graph:

$$Q = \frac{1}{M} \sum_{ij} (A_{ij} - P_{ij}) \delta_{g_i g_j}, \tag{1.4}$$

where M is the total number of edges in the network, A is the adjacency matrix, δ is the Kronecker delta, g_i is the group index of node i (which module the node is in), and P_{ij} is the probability of a connection between nodes i and j if we randomized the edges respecting node degrees. For our network, we can analytically compute (section A.1):

$$Q = \frac{1}{2} \cdot \frac{1-p}{1+p}. \tag{1.5}$$

This varies from $Q = 0$ when $p = 1$ (all nodes connected, so no modularity) to $Q = 1/2$ for $p = 0$ (no connections between the sub-networks, so perfect modularity). Note that for g equally sized groups in a network, the maximum value that Q can attain is $1 - 1/g$.

1.3.4 *Metrics*

In order to compute a metric of functional modularity at the network level, we start by constructing a measure of the strength of the relationship between the activity of a module m and a sub-task (predicting digit k). We use three such measures (described below) based on performance or correlation. We write this as $\mathcal{M}(m, k) \in [b, 1]$ where a value of b means no link (no performance or correlation) and a value of 1 means a strong link (perfect performance or correlation). b represents the base value of a metric, and indicates a null relationship between an agent and a sub-task. We use this (metric-specific) value to normalize properly each metric, resulting in the normalized module-and-digit specific measure:

$$\bar{\mathcal{M}}(m, k) = \frac{\mathcal{M}(m, k) - b}{1 - b} \in [0, 1] \tag{1.6}$$

From this measure that is specific to a particular module and digit (m, k) we construct a module-specific measure of specialization by computing the difference

$$\mathcal{F}^m = \bar{\mathcal{M}}(m, 0) - \bar{\mathcal{M}}(m, 1) \in [-1, 1]. \tag{1.7}$$

This measures how much more linked the module m is to one digit rather than the other. This has value 1 if the module is fully specialized on digit 0, or -1 if it is fully specialized on

digit 1. Finally, the overall specialization of the network is maximum if both modules are fully specialized on opposite digits:

$$\mathcal{F} = \frac{|\mathcal{F}^0 - \mathcal{F}^1|}{2} \in [0, 1] \tag{1.8}$$

With this general structure in place, we define the three core measures as follows:

$\mathcal{M}_{pr}$: **Module Probing** (fig. 1.1C.1). After training on the global task above, we freeze the model and re-train a separate neural network to classify one of the two digits from the outputs of only one of the modules m. With this measure, the module m is specialized on digit k if the accuracy of classifying that digit $\mathcal{M}_{pr}(m, k)$ is high, and the accuracy of classifying the other digit is low. This corresponds to property 1 from the introduction and is also similar in spirit to the information bottleneck principle (Tishby et al., 2000) in measuring how much irrelevant information has been discarded. The base value b is chance accuracy (0.1 when computing MNIST digits).

$\mathcal{M}_{ab}$: **Module ablation** (fig. 1.1C.2). Following the concept of separate modifiability (property 2), we once again freeze the network after training on the global task, but this time use a readout network common to both modules to predict the digits. The ablations metric measures the loss of performance when masking an entire agent's state on a given sub-task. We say the module m is specialized on digit k if its ablation critically impairs classification on this digit but not the other. The base value b is once again chance accuracy.

$\mathcal{M}_{cr}$: **Hidden state correlations** (fig. 1.1C.3). Finally, following the concept of domain specificity (property 3) we aim to understand the input-state relationship of modules of the network. Specifically, we measure how changing one decision variable (digit) while holding the other fixed impacts each module's state. We say the module m is specialized on digit k if there is a higher Pearson correlation coefficient between hidden states resulting from pairs of examples x and x' where digit k of the two examples is the same, compared to the base correlation. This base correlation is the mean correlations of hidden states across all digit pairs (not holding one fixed), and is also used as the normalizing base value b. Recently, work examining the representational similarities of neural nets has shown that because of the inherent non-linearity present in ANNs, a better suited metric to compare multiple networks, or layers of a networks, is the CKA measure (Kornblith et al., 2019). While we agree that this is important when comparing modules that could have the same function with different parametrisations, we are only interested here in measuring the self-similarity of the modules, thus a standard correlation analysis is sufficient.

1.4 RESULTS

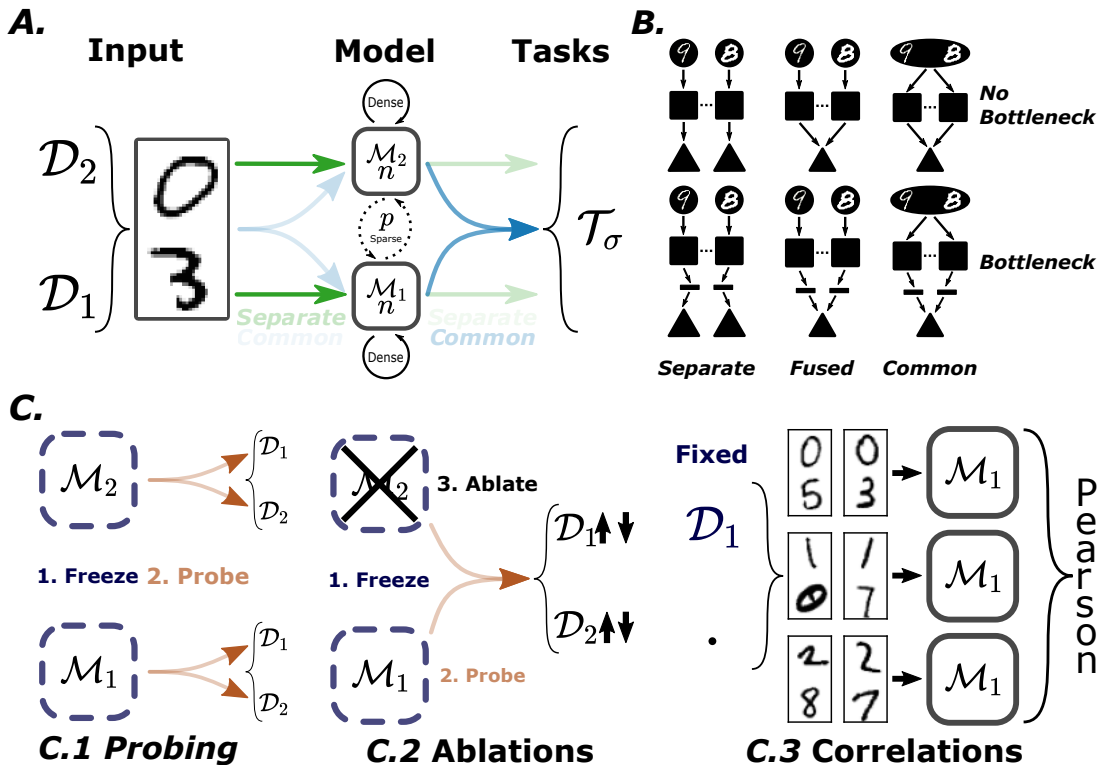


Figure 1.1: Summary of methods. **A.** Schematic of input, model and task. Two digits $\mathcal{D}_1$ and $\mathcal{D}_2$ are fed to the network via either shared input weights (in which case each module sees each digit) or separate (in which case each module sees only a single digit). Two recurrent neural network modules of size n then process the inputs, and communicate via a sparse interconnection containing a fraction p of all n^2 possible connections. Outputs are computed either through shared readout (both modules feed into the same readout) or separate (in which case each module has its own readout, and the decision is based on a max function). A bottleneck consisting of an additional layer of 5 neurons can be placed between the modules and the final readout. **B.** All possible architectural choices. **C.** The three functional specialization metrics: Module Probing, where the trained network is frozen and a separate (probing) network is trained to extract digit identity from each module's hidden state; Module Ablation, where the trained network is frozen and a single network is trained to extract digit identity from both modules' hidden states, after which one is ablated and impact on performance is measured; and Correlation Analysis, where hidden states are correlated holding one digit fixed and varying the other.

1.4.1 *Moderate structural modularity is not a sufficient condition for specialization*

We wish to investigate whether imposing structural modularity is enough to guarantee specialization. To test this idea in a simple and controllable setting, we use an architecture composed of two densely recurrent modules, each receiving its own input (image of an MNIST digit) and producing its own outputs (the *separate-pathway* architecture from fig. 1.1.B). The global task requires each module to recognize the digit presented to it, but also depends on the parity of the other digit. It can therefore be broken down into two sub-tasks (digit recognition) that are able to be solved by the modules individually, but where (idealising somewhat) solving the global task requires the modules to share just one bit of information from the solution

of their sub-task (as it depends on whether or not the parity of the two digits is the same or different). Modules consist of 25 neurons, a regime where the network can solve the task, without being over-parameterized (a parameter choice we will understand in more detail in the next section). Then by varying the fraction of active connections p in the communication layer between modules, we can directly control the structural modularity of the model. For dense intra-module connectivity and a proportion p of active inter-module connections, the standard measure of structural modularity can be calculated as $Q = (1-p)/2(1+p)$ (see section 1.3.3). Test accuracy on this task increases with p (unsurprisingly given that this means more parameters), but is always above chance and below saturation (fig. A.1).

We measured the level of specialization of networks with varying inter-connection sparsity p using three metrics (section 1.3.4) that capture different aspects of functional specialization (section 1.2). We find that even at levels deemed very modular ($Q > 0.4$), the model only shows weak specialization (fig. 1.2.A). We can see how sharply the metrics rise when approaching maximal Q values, suggesting extreme levels of structural modularity are needed for specialization to emerge. The relationship is clearer as a function of p on a logarithmic axis, with specialization naturally dropping when approaching dense communication (fig. 1.2.B). This lets us draw some initial conclusions: in at least one simple type of network, imposing moderately high levels of structural modularity doesn't directly lead to the emergence of specialized modules. Moreover, the metrics we defined show similar trends, indicating that we are indeed capturing something meaningful. For the remainder of this paper, we present results mainly using the module probing metric.

1.4.2 *Environmental structure and resource constraints determine specialization*

We have seen that a high level of structural modularity isn't a sufficient condition for the emergence of specialization. We now systematically analyze the effect on specialization of the choices made in the architecture and environment of our toy model, to understand why and how specialization does emerge. We find that three main categories of effects interact to influence the emergence of specialization.

First are environmental effects: In the previous section, we explicitly designed the data to have a perfectly separable structure. However, features are rarely perfectly independent in real environments (a horse is more likely to have a green background, while a car is more likely to have a gray background). To model these complex dependencies in a simple way, we vary the covariance matrix of the distributions of digits between sub-tasks, essentially modifying the probability of seeing the two digits being equal, $\mathcal{D}_1 = \mathcal{D}_2$. We find that high covariance c prevents specialization (fig. 1.3 bottom left, or top right of each sub-figure), as knowing one hidden variable is already highly informative of the other. In Supplementary fig. A.3, we can also see that when the inputs to the two modules are drawn from separate datasets (as is the

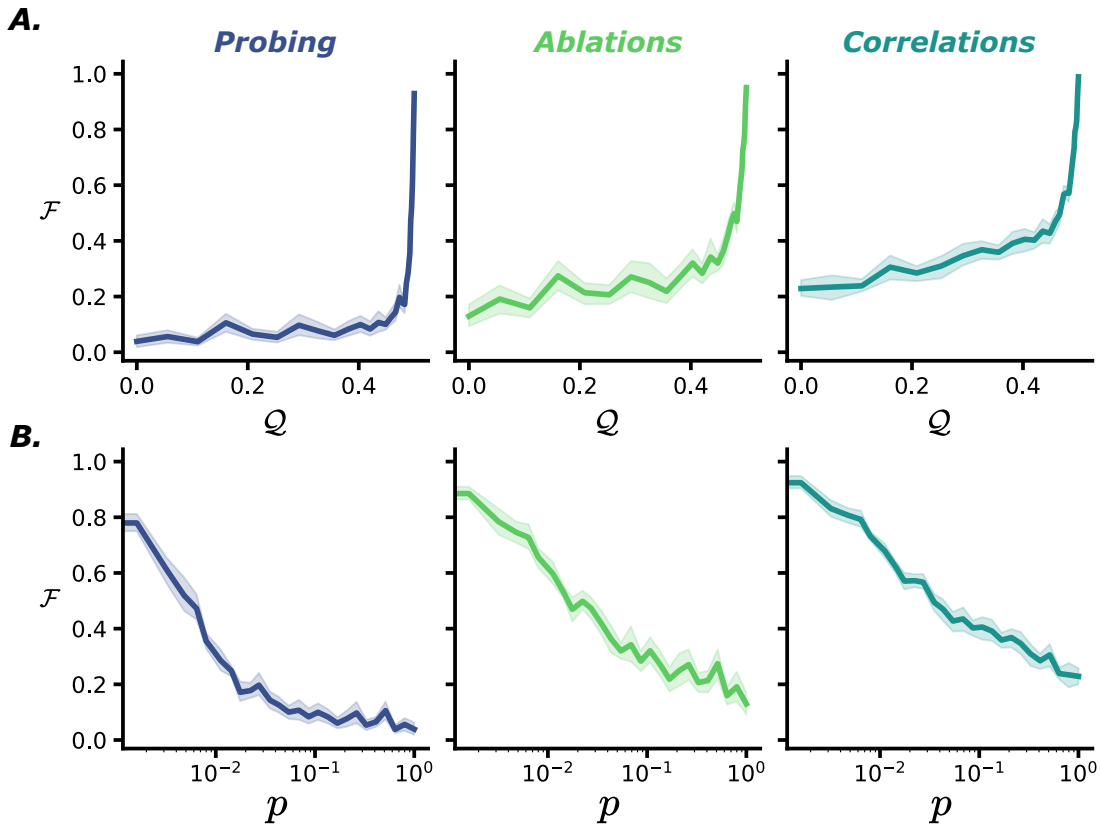


Figure 1.2: Levels of functional specialization of networks with (A) varying modularity Q, or equivalently (B) inter-module connectivity p, measured by three different metrics (columns). All metrics indicate a similar trend, sharply rising only at extreme levels of Q modularity. Data are presented as mean values with a shaded standard error envelope.

case when training with E-MNIST section 1.3.1) modules show stronger specialization. This would be the case in a multimodal setting, such as audio-visual integration. It makes sense that when sub-tasks have different input distributions, this should promote specialization, since the knowledge of decoding one wouldn't transfer as much to the other. Moreover, the computational capacity needed to decode both would be higher when sub-tasks are different in nature, changing some of the metabolic cost effects we discuss later on.

Second we look at metabolic cost constraints: To understand how resource constraints could shape the structure-function relationship of neural networks, we vary the number of elements that are costly to maintain or manufacture (in terms of both energy and space): neurons in a module (n) and synapses between modules (proportion p of n^2 possible, or total number $p_s = pn^2$). We then measure the resulting specialization of the networks. Both constraints, when pushed towards their minimal values, lead to specialization arising, but the extent of which depends strongly on the architectural choices (as discussed in the next point). We can also see that the precise interactions between environmental (c) and metabolic (n, p) variables strongly depend on the choice of the model architecture (fig. 1.3).

Third, we investigate architecture choices: The effect of architecture choice is more subtle. A few salient points include:

1. Input scheme: We find that having separate input pathways constitutes a strong structural prior, leading to higher functional modularity, whereas having a shared input pathway lowers it substantially (fig. 1.3.A/B vs fig. 1.3.C). This intuitively makes sense, as modules are forced to "work with what they've got". When presented with different inputs, the only source of collapse of specialization comes from the information leakage introduced by the interconnections. As such, the main factor driving specialization, in that case, is the number of synapses p_s (fig. 1.3.A top left). Nonetheless, having a shared input does not prevent the appearance of a specialization pattern, albeit at a reduced scale. We find that in that case, the significant parameter is the size of modules, ie the number of neurons they're composed of (fig. 1.3.C top left). Modules are forced to specialize when arriving at the threshold at which a single module does not have the computational capabilities to predict both digits (even when having both as input).
2. Bottleneck: Imposing a narrow bottleneck before the modules' output also lead to higher specialization overall (fig. A.3). This may be because introducing a bottleneck on the output of the modules limits the possible computational role of the output layer itself, and forces the modules to do the majority of the work, which is a prerequisite for seeing specialization in the modules.
3. Output scheme: With separate outputs, the modules compete for which module determines the global readout through a max function. In that case we see a quantitatively lower specialization (fig. 1.3.A vs fig. 1.3.B). This could be explained by the *decision-making* dynamics of modules competing for the readout (a simplified demonstration of this effect is showed in Supplementary fig. A.2). In this structural configuration, networks often resort to having a main *decision-making* module. Although the task is designed to be solvable in a modular fashion, having such a module take all decisions means it's able to predict both digits and thus displays less specialization, when it comes to our metrics at least. One could nevertheless argue that resorting to such a separation of function (decision-making vs context-providing modules) is in its own right a form of specialization, albeit one that is not picked up by our metrics. Finally, having a common readout seems to have a coordinating effect leading to modules being more specialized across the board, alleviating the necessity of extreme resource constraints to see the emergence of specialization.

Results from the complete parameter sweep are shown in the supplementary materials (fig. A.3).

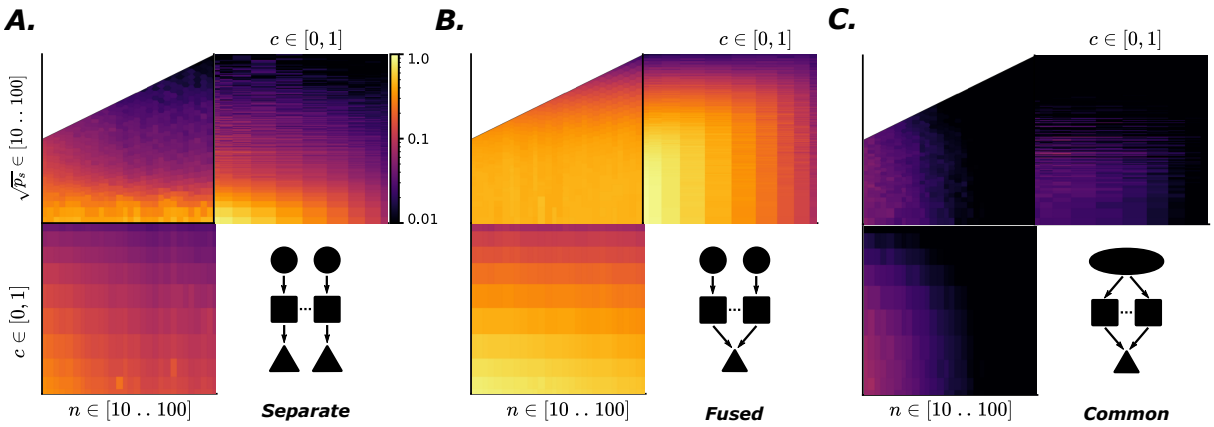


Figure 1.3: Functional specialization of networks with varying module size n, number of active synapses in the communication layer p_s, in an environment where input variables present covariance c. Each image shows functional specialization when varying two of these parameters (averaged across the third one), with color indicating the degree of functional specialization on a log scale, cut off at a mimal value of 0.01. Results are shown for three different architectures: (A.) separate pathways, (B.), fused pathways, and (C.) shared pathway.

1.4.3 *Function specialization can vary dynamically*

Functional specialization is usually implicitly understood as a static property, i.e. a module is functionally specialized or not. Here, we investigate this assumption by quantifying specialization at every time step (fig. 1.4), in the *fused pathway* architecture. We find that when presented with static inputs, the time course of specialization mainly depends on the timing of inter-module communications (fig. 1.4.A). When communicating at high bandwidth (p), the specialization of modules drops significantly and immediately after modules start to communicate. This collapse happens immediately, with no continual leakage over time, i.e. no further reduction of specialization after the initial drop. We hypothesize that this is a result of the static nature of inputs, where all the information is readily available at every time step and so there is less advantage to ongoing communication after the initial burst (the only benefit of maintaining active communication is to free up working memory resources).

To confirm this intuition based on the static nature of inputs, we introduce dynamic structured noise (section A.5.1). With a noisy input, additional information about the underlying signal is introduced at each time step, giving the network an extra reason to engage in ongoing communication. We use two possible noise levels (fig. 1.4.B). This time we observe a continual decrease of specialization over time, in highly interconnected modules (i.e. the metric keeps dropping after the initial drop). This is especially true when decoding highly noisy versions of the inputs, meaning there is an increased advantage to ongoing communication.

Finally, we introduce stochasticity in the input dynamics by having individual digits be turned on from a purely noisy input at random times (section A.5.2), but with inter-module communication always on (fig. 1.4.C). We find that specialization dynamics follows input

dynamics. When neither digit is switched on, neither module specializes (first time-step of every sub-plot). Once digit 0 (arrow upward) is turned on, module 0 starts to specialize on that digit, and similarly for digit 1 (downward arrow) and module 1. When only one digit is turned on, the other module also starts to specialize on that digit in the absence of any other input to specialize on. Once both digits are switched on, modules start by specializing in their respective digits, before the eventual drop in specialization over time. Specialization is more fluid in more densely connected networks, moving towards the non-primary digit faster in the absence of a primary digit input, and also decaying to zero faster once both digits are present (as in previous results). Intuitively, this can be summarized as specialization following the total amount of information received from the two digit sources (since sparser connectivity means lower bandwidth).

1.5 DISCUSSION

We used a carefully controlled toy network to investigate the relationship between structural and functional modularity. In doing so, our first challenge was to define modularity. For structural modularity, we simply used the popular graph theoretic Q metric (Newman, 2006). For functional modularity, or specialization, there is no similar consensus. Our first main result is that we found that three different metrics inspired by classical definitions of modularity gave broadly compatible results in our setup, suggesting that even though functional modularity is difficult to define and the subject of some contention, there may be a meaningful underlying concept. However, we also found that there is a dynamical aspect which is not captured in these definitions of functional modularity and which may turn out to be critical to understanding information flows in more complex networks and environments. Functional specialization should be understood in terms of states, or trajectories, with modules varying in their specialization depending on input and module communication dynamics. This supports recent developments in neuroscience that have started to rethink the causal narrative behind brain functions in a more dynamical manner and how it is related to a modular organization (Bassett et al., 2011; Shine et al., 2016), for example looking at temporal contributions of elements in a system (Fakhar et al., 2023).

Our second main result is that imposing some degree of structural modularity on a network is not, in general, sufficient to induce functional specialization, even in a toy setting seemingly ideal for the emergence of specialization. We conclude that (a) in machine learning and neuromorphic computing applications, if we wish to encourage the emergence of functional specialization, structural modularity could prove insufficient, and (b) in neuroscience, if we observe structural modularity, we should not necessarily conclude that these modules will be functionally specialized. One limitation on this conclusion is that we have relied on the well-established Q-metric from network theory. This metric is widely used in connectomics

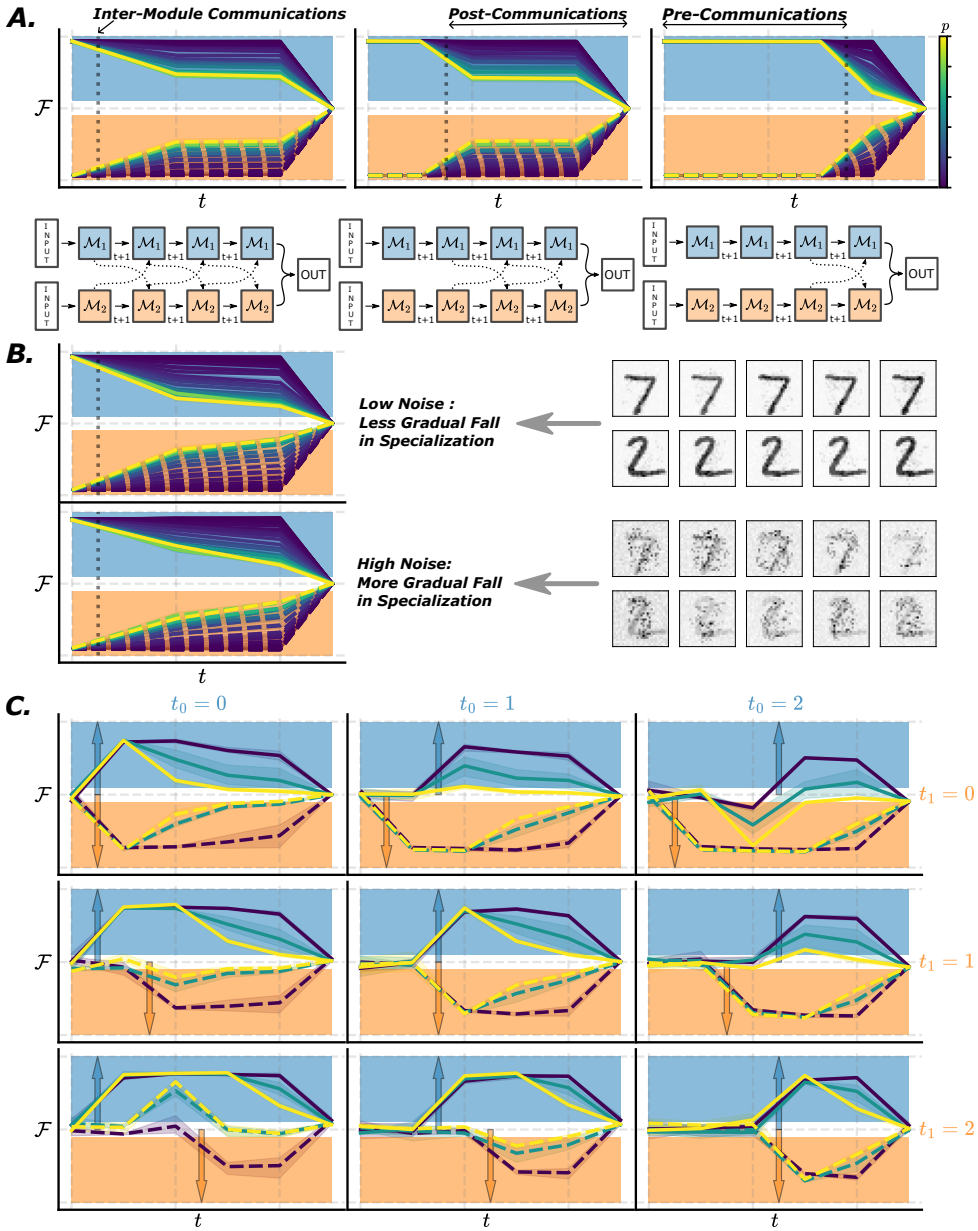


Figure 1.4: Specialization dynamics. **A.** Specialization of different parts of the network at different time-steps. The network unrolled through time is shown below, with inputs presented to modules at time $t = 0$ at the left hand side, and subsequent time steps moving towards the right finishing in the readout layer at the final time step. In the plot above is shown the specialization of the corresponding module at a given time, where specialization close to 1 means full specialization on digit 0, close to -1 means full specialization on digit 1, and close to 0 means no specialization. The dark dashed lines in both the network diagram and the plot show the timing of inter-module communication. Each colored line represents networks with different inter-connection levels p. We vary the communication timings in the inputs, and find specialization collapses when communicating at high bandwidth (p high). **B.** With dynamic noise in the inputs, specialization drops continually, especially in high noise setting. **C.** With stochasticity in the input dynamics (digits 0 and 1 turned on at t_0 and t_1 respectively, indicated by colored arrows), specialization dynamics closely follows the inputs' dynamics. Networks showed with 1 (dark blue), 10 (teal) and 100 (yellow) active inter-module connections. Data are presented as mean values with a shaded standard error envelope.

research, but may not be the best measure of structural modularity (although we did not systematically investigate alternatives).

Structural modularity may well be one component of a richer approach to understanding functional specialization, including more complicated architectures, different learning rules, training regimes, datasets and neuron model (for example, a spiking neural network would have very different bandwidth to an equivalent artificial neural network). By systematically varying details of our toy network, we found that as a first step towards such a richer approach, introducing resource constraints was a reliable way to encourage stronger functional specialization. Intuitively this makes sense, as when computational resources are abundant there is no incentive to share or re-use these resources. Having shown that the link from structure to function is not as straightforward as one might think, an exciting follow-up research question would be to investigate the link between functional and compositional aspects of modularity. This link has already proved harder to understand than expected: even when neural networks do attain functionally specialized organization they do not always make compositional use of these modules (Bahdanau et al., 2019; Csordás et al., 2021; Mittal et al., 2022).

To address the profoundly worrying climate emergency, in the light of the rapid growth in energy costs of AI, mechanisms that encourage parsimonious solutions are likely to be increasingly important. Neuromorphic engineering has the potential to take AI and robotics to a new era of scalable and low-power systems, away from the ever bigger and hungrier nature of current trends. Edge devices, equipped with neuromorphic sensors and processors would be a way for intelligent agents to integrate into society effectively and responsibly (Bartolozzi et al., 2022). Such a decentralized, agent-centric perspective thus directly implies a modular organization, and seeing how such autonomous agents would be resource-constrained, the framework we developed would seem to be well fitted to better understanding their dynamics. Moreover, internal agents' dynamics would also be affected: neuromorphic chips function under the same kind of resource constraints as the brain does, with bandwidth and energy consumption being core preoccupations. Neuromorphic hardware specifically designed to implement networks with small-world topologies have also recently been put forward (Dalgaty et al., 2024). Finally, agents equipped with multiple neuromorphic sensors would naturally be suited to a modular or hierarchical organization, composed of unisensory and multisensory areas. The issues discussed in this paper may then prove important as we enter a new era of intelligent neuromorphic autonomous agents.

Our approach has been very handcrafted, with precise control over structural modularity and connectivity in the networks. However, other studies have taken a more emergent approach: (Achterberg et al., 2023; Liu et al., 2023) investigated whether modular properties can emerge directly from the first principle of minimizing connection costs. Spatially-embedded networks, regularized to minimize connection costs while learning, do end up displaying modular and small-world features, but we note that both works had to introduce an additional regularization

or optimization technique to see it emerge. Understanding if, and if so how, structural and functional modularity can emerge from purely low level and naturalistic principles outside of a controlled setup thus remains an open question. The search for such low level principles as a factor driving emergence of higher level concepts is nonetheless very promising. Another recent study shows how regularizing for energy efficiency could lead networks to organize into a predictive coding regime (Ali et al., 2022), and is a good example of how taking into account physical substrates and constraints can lead to a better understanding of networks' function.

We investigated relatively small networks, solving simple tasks. This enabled us to separate the multiple factors influencing modularity, and to thoroughly explore the parameter, architecture, and environment space. What's more, the simplicity of our setup makes for an ideal proving ground for concepts of modularity: if the current static definition of specialization already shows its limits here, it would then seem too simple to describe complex and dynamic real-world systems. However, the opposite argument could also be made, suggesting that such a simple setup prevents us from seeing the emergence of functional specialization. Other works have taken more holistic approaches, studying both structural and functional modularity in trained networks at the same time (Casper et al., 2022; Filan et al., 2021; Hod et al., 2022). Such an approach could reveal even further de-correlation of function and structure, as in (Casper et al., 2022; Hod et al., 2022) where function is shown to emerge even without modular structure. In that sense by focusing only our efforts, and the scope of our metrics, on the two structurally defined modules we could already be missing functionally specialized yet decentralized structures. Further extensions of this work could also take into account multiple modules and tasks, and introduce varying environments. Indeed, such environments have been shown to favour modular solutions (Ellefsen et al., 2015; Kashtan et al., 2007), and could also lead to different time-scales in specialization dynamics (life-long learning). Not only that, but it has been shown that multi-task learning can drive the emergence of modularity (Dobs et al., 2022; Yang et al., 2020), and as such would be interesting to investigate. In that context, the concept of systematic generalization is of crucial importance, and the ability for an agent to recombine and reuse previously acquired knowledge could prove decisive to its survival. Such an ability being better enabled through modular architectures (Bahdanau et al., 2019; Goyal et al., 2020; Kirsch et al., 2018), we can see how the necessity for generalization could also promote modular architectures to emerge (although this has yet to be shown, as discussed above). Finally, we only looked at the limited learning rule of supervised learning with backpropagation, and other learning methods could potentially lead to different results, as suggested by Bakhtiari et al. (2021).

When it comes to the brain, although the concepts of structural and functional modularity are useful and well established, researchers are increasingly studying what this means in a networked, dynamic, and entangled way (Bassett and Sporns, 2017; Pessoa, 2022). The brain is a messy, complicated system, and we may not find clean and clear-cut concepts of modularity to apply directly. However, using artificial neural networks does give us a clean and controlled

setup to explore these concepts. We can subsequently take these methods and apply them in more complex, uncontrolled environments, including artificial systems working with real world data, or biological systems. For example, a combined approach using artificial neural network decoders of electrophysiological recordings could enable us to measure specialization with the definitions introduced here. We conclude that using ANNs as a testing ground to develop more rigorous conceptions of modularity which can then be applied to experimental data could be a very fruitful approach for both neuroscience and machine intelligence.

2

SPATIAL AND NEUROMORPHIC PRIORS FOR COMPOSITIONAL LEARNING

Narrative Preamble

The previous chapter demonstrated that structural modularity does not automatically beget functional specialization; rather, it requires the tightening of specific constraints (bandwidth, computational power) to "force" the system into a specialized regime. If constraints are the engine of modularity, what is the most fundamental constraint available to a physical system? This chapter investigates to what extent *Space* itself might be that constraint. We move our study to spatially embedded systems, effectively activating the ”Brain Economy” loop (see section 0.5.7). We introduce a mechanistic framework that applies the strict physical penalties of spatial or neuromorphic substrates, specifically the cost of long-distance communication, to recurrent neural networks. We investigate how these **Proximate Pressures (PP)** force the emergence of sparse, modular topologies (**PS**) that are not merely energy-efficient, but functionally superior, showing that these spatially-constrained networks get closer to unlocking true compositional generalization.

Publication Context

This chapter represents an ongoing effort with Melikas Payvand’s team at the Institute of Neuro-informatics (collaborators: Jimmy Weber, Yassine Benchakroune) in Zurich. It is not yet published.

Contributions

This chapter represents a collaborative effort led by the author. The author is responsible for the primary design and implementation of the experimental pipeline, the execution of all simulations, the data analysis, and the writing of the manuscript. The conceptual genesis of the main dataset emerged from pivotal discussions with Dan Goodman. We acknowledge Jimmy Weber for establishing the foundational concepts for extending DEEP-R into a hardware-software co-design tool and for providing the initial Mosaic Neuromorphic codebase, which was turned into a spatial embedding and integrated into the training pipeline by the author. Finally, both Melika Payvand and Dan Goodman provided overall supervision and guidance throughout the project while Jimmy Weber and Yassine Taoudi-Benchekroun provided valuable technical insights, critical feedback on the results, and engaged in fruitful discussions that shaped the direction of the study.

2.1 INTRODUCTION

2.1.1 *The Combinatorial Imperative: From Monolithic Mapping to Compositional Reasoning*

A central challenge confronting any intelligent system—whether biological or artificial—is the problem of combinatorial explosion. The physical world presents itself not as a pre-enumerated list of monolithic states, but as a composite reality constructed from a finite set of primitives interacting through bounded rules. While atomic elements may be enumerable, the space of their potential configurations is effectively infinite, a phenomenon famously described as "the infinite use of finite means" in linguistics (Humboldt's classical observation on language famously formalized by Chomsky, 1965). Consequently, an agent modelling its environment cannot rely on sheer memorization of state-action pairs.

Consider a domain with N atomic concepts combinable into structures of length L, yielding a search space $\mathcal{S}$ of size $|\mathcal{S}| = N^L$. If a learning system treats every possible combination as a completely independent entity, its sample complexity—the amount of training data required—grows explosively with the combinatorial possibilities (N^L). This creates a fundamental barrier: as domains become more complex, covering the space of combinations becomes practically impossible. This challenge is intrinsically linked to the *curse of dimensionality*: in high-dimensional spaces, data points become sparse, and covering the space uniformly requires exponentially more samples. This is not merely a statistical inconvenience but a barrier to generalization; without exploiting structure, generalization to novel combinations is impossible by construction. Indeed, it has been long-known that general function approximators are, by virtue of their very generality, notoriously data-hungry (Geman et al., 1992).

The solution exhibited by biological intelligence—and arguably a central objective of artificial intelligence—is *systematic compositionality*: the algebraic capacity to understand and produce infinite behaviours from finite primitives. A compositional learner operates fundamentally differently: rather than memorizing combinations, it masters N atomic concepts and their rules of interaction, building novel combinations by *reusing* these learned blocks. To visualize the impact on complexity, consider a function that can be completely factorized into atomic components: $f(x_1, ..., x_L) = f_1(x_1) + ... + f_L(x_L)$. A learner exploiting the underlying factorized structure of this data need not memorize the global look-up table. Instead, it can rely on L look-up tables of N elements each, reducing the complexity from N^L to $N \times L$. While most learning problems are not purely factorizable, and most learning systems are not pure look-up tables, this intuition highlights a key principle: *compositionality makes learning problems tractable*.

Mathematically, this reflects a principle from statistical learning theory: a learner's data requirements depend on the capacity (or complexity) of the hypothesis space it searches (Blumer et al., 1989; Vapnik and Chervonenkis, 1971). A monolithic learner must search a space capable of representing an exponential number of arbitrary possibilities. In contrast, a

compositional learner constrains its hypothesis space to respect the data's structure, potentially reducing the sample complexity from exponential to polynomial scaling. Empirically, this principle manifests in the dramatic efficiency gap between human and machine learning. As Lake et al. (2016) observe, humans master novel visual characters from a single example and grasp novel video games in minutes, whereas contemporary neural networks often require tens of thousands of instances. This disparity suggests that compositionality—likely combined with causal reasoning and learning-to-learn—is a key prior in enabling such sample efficiency.

2.1.2 *The Landscape of Compositional Generalization*

Generalization in machine learning typically relies on the assumption that test sets are drawn i.i.d. from the training distribution. Compositional generalization fundamentally breaks this assumption, presenting an *out-of-distribution* (OOD) challenge: models must recombine known components (atoms) into novel structures (compounds) never encountered during training (Hupkes et al., 2020). This capability is often categorized into three dimensions: *systematicity* (recombining known words into new valid sentences), *productivity* (extending reasoning to sequences longer than those seen in training), and *substitutivity* (robustness to synonymous components) (Fodor and Pylyshyn, 1988; Lake and Baroni, 2018). While humans excel at these tasks, standard neural networks frequently fail, defaulting to bag-of-words heuristics or superficial pattern matching (Keysers et al., 2020).

Early benchmarks like SCAN (Lake and Baroni, 2018) and COGS (Kim and Linzen, 2020) were pivotal in exposing these failures when syntactic structures shifted. Some of the field has recently moved toward strictly *algorithmic* reasoning tasks, such as Pointer Value Retrieval (PVR) (Zhang et al., 2022). In this paradigm, the input sequence functions as a buffer where tokens play dual roles: a number is both a data value and a potential address (pointer) for the next retrieval step. The task effectively demands the decoupling of *data values* from the *control flow*, as the model must interpret the first token as an instruction for where to look next. Extensions like Conditional PVR (C-PVR) (Abnar et al., 2023) escalate this complexity by introducing dynamic halting conditions—such as "follow the pointers until the value decreases." This forces the network to support *adaptive computational depth*, dynamically executing a variable chain of reasoning steps determined solely by the input data. While these benchmarks are invaluable, they often serve as monolithic pass/fail tests rather than diagnostic tools for understanding the *emergence* of compositional ability.

Given that complex problems naturally decompose into sub-routines, a compelling architectural hypothesis is that *structural modularity* is a deeply linked to compositional learning (Lepori et al., 2023). This intuition—that network structure should mirror task hierarchy—has driven the development of Compositional Neural Architectures (a more complete taxonomy of which can be found in Sinha et al., 2024). Pioneering works like Neural Module Networks

(NMNs) (Andreas et al., 2017) explicitly assemble command-specific networks from pre-trained modules to match linguistic structures, while neuro-symbolic hybrids (NeSS) (Chen et al., 2020) merged neural-networks with symbolic logic enabling robust compositions. Similarly, Kuo et al. (2021) demonstrated on the gSCAN benchmark (Ruis et al., 2020) that modular architectures allow for greater interpretability, theoretically enabling the model to assemble previously learned components for novel tasks. However, this explicit engineering comes with limitations: Subramanian et al. (2020) observed that enforcing rigid modularity often results in modules that are not "faithful" to their designated concepts, despite high task accuracy.

2.1.3 *The Mechanistic Gap: Controlling Complexity and Emergence*

Moving beyond observational performance to understand the *mechanisms* of generalization led us to a seemingly straightforward question: *would more modular networks inherently generalize better on more compositional tasks?* In other words, to what degree does structural modularity correlate with compositional competence? Yet, attempting to empirically probe this hypothesis reveals a dual methodological barrier in both current datasets and modelling paradigms.

The Data Challenge: A lack of defined, tunable complexity. To rigorously link modularity to generalization, one requires a "compositionality slider", a way to independently adjust the difficulty of the underlying structure without altering other statistical properties. The field lacks formal definitions of what renders a dataset "more" or "less" compositional in a way that allows us to probe specific model capabilities in isolation. Additionally, existing benchmarks often inadvertently couple *algorithmic complexity* with *statistical distribution*. For instance, manually increasing the reasoning depth in multi-hop tasks (Abnar et al., 2023) (an effect the authors themselves albeit did not investigate) can inadvertently alter input token distributions (e.g., changing the frequency of pointer values).

The Modelling Challenge: Engineering vs. Emergence. On the modelling side, while the necessity of compositionality is established, the mechanisms by which it might *emerge* from unstructured substrates remain opaque. Recent successes largely rely on highly engineered, handcrafted architectures (as we've just discussed) or massive over-parameterization (Kaplan et al., 2020; Redhardt et al., 2025). While these approaches are valuable and effective, they could obscure the links between general inductive biases and compositional ability. We lack a fundamental understanding of whether compositional reasoning can emerge from *unstructured* networks without explicit, hard-coded modules. Furthermore, rigorous analysis is hindered by the difficulty of defining fair baselines: in highly engineered networks, specific architectural design choices often outweigh the contribution of general computational capabilities (such as parameter count). To understand the origins of compositional learning, we require a controlled setting where inductive biases can be varied while maintaining fair comparisons against strict baselines.

2.1.4 *Biological Constraints as Inductive Biases*

Biological systems offer instructive precedent. The brain is a spatially embedded network subject to severe metabolic and physical constraints, occupying finite volume with connections consuming space and energy. The "brain economy" hypothesis (Bullmore and Sporns, 2012*a*) suggests that topological features like modularity, small-worldness, and hubs emerge as evolutionary byproducts of pressure to minimize connection costs while maintaining processing efficiency. Connection costs scale with distance and volume, creating selective pressure favouring short connections. This distance-based constraint, far from being merely limiting, may actually in turn drive modularity and evolvability (Clune et al., 2013; Ellefsen et al., 2015; Jacobs and Jordan, 1992; Mengistu et al., 2016).

Recent work has revived these insights in deep learning contexts. Achterberg et al. (2023) demonstrated that Spatially Embedded Recurrent Neural Networks (seRNNs), with neurons embedded in metric spaces and regularized by distance-penalized connection costs ($L = L_{task} + \lambda \sum |w_{ij}| d_{ij}$), spontaneously develop modular, small-world topologies resembling biological connectomes while exhibiting functional advantages. Similarly, Abnar et al. (2023); Khona et al. (2023); Zhang et al. (2025) showed that wiring cost control improves performance across diverse tasks and architectures, identifying optimal wiring cost ranges achieving Pareto-optimal balances of performance, modularity, and clustering. Modular networks born of wiring constraints appear to learn better and reconfigure more easily than entangled monolithic ones, suggesting a powerful, biologically grounded inductive bias: *geometry*.

However, a critical distinction exists between structural modularity (physical node clustering) and functional modularity (task factorization into independent sub-routines). In Chapter 2, we showed that networks can be highly modular connectivitywise (high Q-metric) yet functionally entangled, with single modules participating in multiple unrelated sub-tasks. We identified two conditions that needed to be met, in this particular setup, in order for specialization to in fact arise: *environment separability* (factorizable tasks) and *resource constraints* (scarcity forcing efficient allocation rather than distributed redundancy). If we manage to solve both the *data* and *modelling* we've just described, exploring space-constrained neural networks in a compositional environment could lead to both conditions being met.

2.1.5 *Contributions and Key Findings*

We introduce a dual framework designed to probe the emergence of compositional ability under varying spatial constraints. Our contributions and findings are summarized below:

- **A Tunable Testbed for Compositional Reasoning:** We propose a novel meta-dataset that reformulates the problem of pointer-value retrieval into a *grid navigation task*. Conceptually,

this can be visualized as an agent navigating a maze of rooms: in each room, the agent must solve a local puzzle to retrieve a tool (an atomic function) and a map to the next room (a pointer). This allows us to disentangle two previously confounded dimensions of difficulty: *sequential complexity* (the depth of the reasoning chain) and *combinatorial complexity* (the size of the library of subtasks). This enables us to thoroughly test generalization in a controlled setting where difficulty can scale without altering input statistics.

- **Injecting Physical Constraints into Differentiable Models:** We develop a hardware-software co-design framework capable of training sparse Recurrent Neural Networks (RNNs) under strict physical penalties. By embedding neurons into metric spaces—ranging from generic geometries (grids, cubes) to specific neuromorphic hardware layouts—we can rigorously compare "unconstrained" baselines against networks subject to the "brain economy" pressures of minimizing wiring distance.
- **Spatial Constraints Drive Generalization and Efficiency:** We demonstrate that spatially embedded networks significantly outperform their non-spatial counterparts in both systematic and length generalization. Crucially, this advantage is most pronounced in high-width settings, suggesting that spatial priors are particularly valuable when agents must navigate large libraries of atomic primitives. Furthermore, these physical constraints fundamentally alter training dynamics, leading to significantly improved sample efficiency and earlier generalization.
- **The Structure-Function Gap:** While spatial penalties successfully force the emergence of highly modular network topologies (clusters of physical connections), we find this does not translate directly into functional specialization. Structural modules do not necessarily become "specialists" for specific sub-tasks, implying that spatial priors drive performance through global organizational properties rather than by isolating distinct functional sub-routines, contrary to prior findings in the field.

2.2 METHODS

Investigating compositional generalization's causal mechanisms requires two tools: a problem domain where compositional complexity can be defined and adjusted, and a modelling framework where priors can be injected under controlled constraints and compared against fair baselines. We detail both below.

2.2.1 *The 2D-PVR Framework: A Tunable Compositional Testbed*

Motivation

Building on the landscape established previously, we recognize that "difficulty" in compositional reasoning is not monolithic. We decompose the problem space into four tunable axes. Our goal will then be to design a dataset (or meta-dataset) allowing us to precisely tune each aspect individually in order to investigate correlations and causal links with model properties of interest.

- *Symbolic complexity* (control flow): the algorithmic core difficulty of the task, distinguishing simple function chaining from true programmatic reasoning.
- *Sequential complexity* (depth): the number of reasoning steps required to solve a problem, creating dependency chains where each step's output becomes the next input. This test models' ability to maintain stable forward-propagated state and extend learned algorithms to unseen sequence lengths (length generalization). A model relying on of sheer memorization would need an increasing number of resources to learn deeper examples, while a model capturing the algorithmic core of a task would generalize directly.
- *Combinatorial complexity* (width): the number of necessary sub-routines needed to solve a problem. While all datasets aren't always so clear-cut, the idea of compositionality implies that there is, at least in some form, some atomic sub-functions that need to be combined to make a greater whole. Wider problems imply larger combinatorial operation spaces. Depending on which amount of the combinatorial space is covered during training, this can test for systematic generalization: the ability to recombine known sub-routines in novel configurations never explicitly presented before.
- *Atomic complexity*: individual function (primitive) difficulty, independent of larger program overhead.

Sequential Compositional Tasks

In order for us to grasp the implications of dividing up complexity in this way, let's consider a baseline. In a "straight line" task, a learner needs applying specific transformations from instructions passed at each time step: $y = f_{k_T} \circ f_{k_{T-1}} \circ \cdots \circ f_{k_0}(x)$ (fig. 2.1). Such as task can feature a pool of many *atomic* sub-functions ($\{k\}_i \in [\![1, N]\!]$) chained for many steps T, meaning it possesses dimensions of width and depth. While possessing those dimensions, it lacks symbolic challenge (execution paths are fixed and explicit), requiring only correct operation sequence application without decision tree navigation or complex state maintenance (beyond forward transformation).

To introduce the necessary algorithmic complexity—specifically, the dimension of control flow—we turn to **Pointer Value Retrieval (PVR)** (Abnar et al., 2023; Zhang et al., 2022). PVR

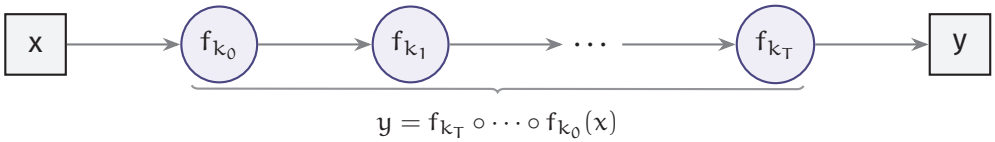


Figure 2.1: A baseline sequential compositional task: in such a setup, outputs a computed through chained compositions of transformation drawn from a library. While it possessed dimensions of depth (T) and width $|\{f\}|$, it lacks algorithmic complexity.

replaces the explicit instruction set with indirect addressing, where the data itself dictates the control flow. In a PVR task, the model must look at a specific "pointer" index, retrieve its value, and use that value as the address for the next step. This forces the model to perform dynamic, data-dependent reasoning rather than static sequence processing. However, this increase in symbolic complexity comes at a significant cost: we sacrifice the clean definition of compositionality found in the baseline. First, the *functional* composition disappears. We are no longer chaining semantic sub-routines (e.g., $f \circ g$), but simply hopping indices. Consequently, defining distinct "sub-tasks" or measuring "combinatorial width" becomes ambiguous. Second, and perhaps more critically, standard 1D PVR suffers from a coupling between *depth* and *data distribution*. In a 1D array, creating a reasoning chain of length T requires a specific sequence of values (e.g., index 0 points to 1, 1 points to 2, etc., as seen in fig. 2.2). As we increase the target depth, the distribution of values within the input sequence must change to accommodate the longer chain. This effectively confounds *algorithmic generalization* (learning to reason deeper) with *distributional shift* (learning to process different data statistics). Furthermore, as chain depth grows, more "slots" in the sequence are consumed by the signal, leaving fewer slots for noise/distractors, which ironically can make deeper examples easier to identify than shallow ones.

We therefore require a framework that bridges this gap: one that retains the algorithmic, data-dependent control flow of PVR, but restores the clean, independent axes of compositional complexity found in linear chaining.

The 2D Paradigm Shift

We address these limitations by proposing a 2D grid navigation framework (fig. 2.3). Moving from 1D tapes to 2D grids introduces mechanisms decoupling *syntax* (reasoning paths) from *semantics* (values and functions). Consider an analogy: imagine rooms (time steps) each presenting multiple puzzles (grid cells), but only one puzzle contains the relevant clue. Solving the correct puzzle reveals an *instruction* (action to perform, e.g., “shift cipher letters by 1”) and a *pointer* (next room’s correct puzzle location). This structure creates paths hidden within noise: wrong early choices lose the reasoning thread entirely, forcing agents to carry forward internal state (pointers to next valid cells) as well as propagate data transformations.

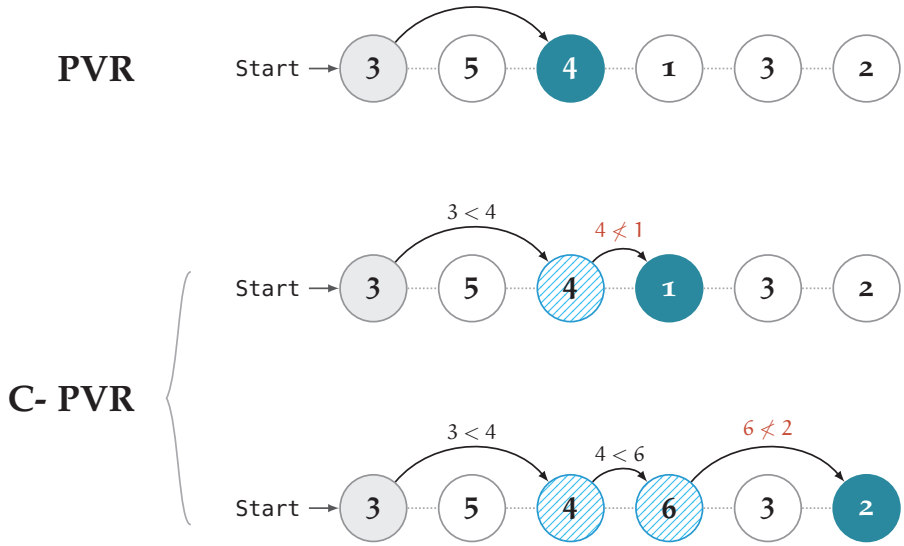


Figure 2.2: Original PVR and C-PVR Datasets. Adapted from Abnar et al. (2023). In the regular PVR task, only one part of the input (in this case, the first element in the sequence) serves as the pointer. However, in the C-PVR, each element can act as a pointer. The task then becomes to recursively retrieve the values until a condition is met. For this illustration, the condition is to continue as long as the sequence values increases.

This 2D formulation possesses key properties. Grids function as programs that neural networks must "compile" and execute, with identical grid structures applicable to different input vectors, enforcing functional understanding over pure memorization. Every value and instruction appears at every time step, eliminating exploitable statistical regularities (e.g., "small numbers imply long paths"). While tasks require carrying forward information over time, memory required for *control flow* (pointers) is constant and does not scale with sequence length. Models correctly inferring underlying algorithms should generalize to arbitrary lengths automatically, making length generalization an effective probe for symbolic task understanding.

Implementation Details

In our specific implementation, we define atomic libraries ($\mathcal{F}$) as linear transformations over discrete spaces. Libraries consist of N_W unique 5×5 permutation matrices generated in a greedy manner to have sufficient Hamming distances between them (ensuring distinctness) and minimize fixed points (ensuring sufficient mixing). We investigate widths $W \in \{10, 15, 25, 50\}$. Input vectors x_0 and subsequent targets are 5-dimensional vectors, with inputs being drawn without replacement from $\{0, 0.2, 0.4, 0.6, 0.8\}$, (thus all intermediate targets are also permutations of these identical elements).

At each time step t, models observe columns C_t of height H, with each cell containing a *pointer* (one-hot encoding of next row index $p \in [\![0, H-1]\!]$) and an *instruction* (flattened 5×5 function weight matrix). Valid cells at step t are determined by pointers retrieved at $t-1$; all

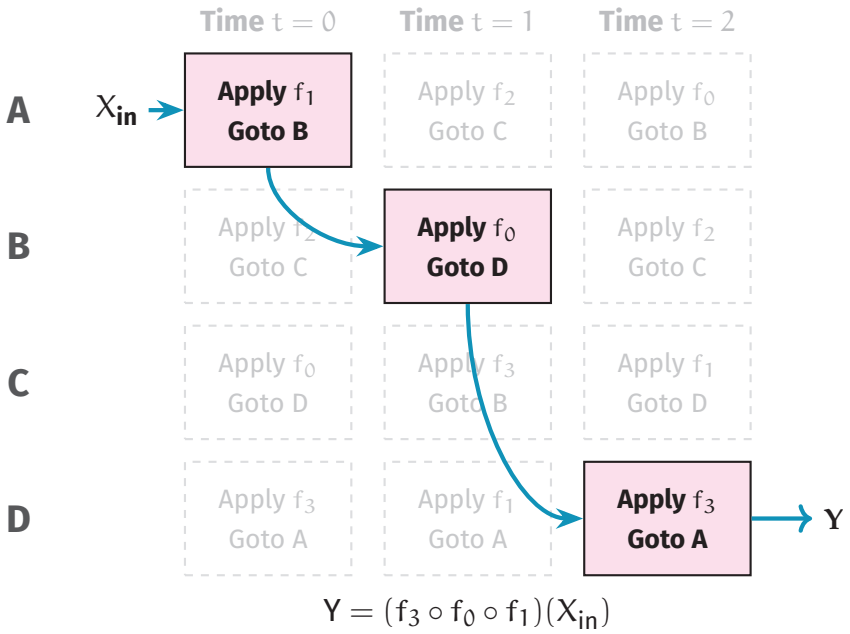


Figure 2.3: Schematic representation of the 2D Grid. Each cell contains a pointer P (next address) and an instruction I (function to apply). The model must navigate the path amidst distractors while chaining atomic functions forward in time to transform an initial vector X.

other $H - 1$ cells serve as distractors. Models must apply valid instructions to internal state y_{t-1} to produce y_t. We investigate grid heights $H \in \{5, 10, 15\}$.

We generate training sequences of length $T_{train} = 10$ steps and evaluate OOD on lengths of $T_{ood} = 20$ steps, computing OOD accuracy only for timesteps $t > 10$. To prevent memorization of specific path combinations, we empirically verify the validation sets consist of unique paths unseen during training (this is ensured by combinatorial explosion even at lowest height parameters). Networks are trained via Backpropagation Through Time (BPTT). To facilitate task logic learning, we employ *random intermediate queries*: losses compute not only at final steps but at random intermediate timesteps, providing gradient signals encouraging step-by-step solution derivation (this effectively simulates training on varying-length examples up to T_{train}).

2D-PVR as a Meta-Dataset

The 2D PVR framework's flexibility enables meta-dataset functionality. Changing specific parameters generates diverse datasets targeting specific capabilities:

- *Symbolic complexity* (control flow and attention): This can vary through pointer logic (absolute addressing vs. relative offsets vs. path-based functions), grid height (distractor density, taller grids require more robust attention for filtering irrelevant information, distinct from function library size), value encoding (one-hot vs. embeddings forcing perceptual sub-routine learning), and intermediate supervision and curriculum (restrict-

ing training to final time-steps forces implicit inference of entire chains, while random intermediate queries provide "glimpses" into the algorithm).

- *Atomic complexity* (function difficulty): This can be adjusted through function library selection (though cleanly defining function complexity remains context-dependent on the precise models trained to solve tasks) and transformation space (larger input dimensionality and different input distribution nature influence atomic operation difficulty, uncorrelated with the pathfinding symbolic logic).
- *Combinatorial complexity* (width) directly scales via library size, measuring compositionality by forcing models to compose larger atomic unit varieties rather than memorizing common patterns.
- Finally, *sequential complexity gap* (depth) arbitrarily extends grid length (T) to test OOD length generalization. Because local statistics (pointer distributions, instruction types) remain constant regardless of length, this probes pure length generalization without confounding distributional shifts. Varying the training vs OOD step-number gap can also test the ability for algorithm inference from relatively short training examples.

2.2.2 *Spatial DEEP-R: Hardware-Software Co-Design*

Motivation

We investigate Recurrent Neural Networks (RNNs) tackling the compositional sequential tasks described above. Specifically, we seek to understand what inductive biases can be introduced "for free" to unstructured RNNs to enhance compositional learning and symbolic reasoning. While prior work demonstrates that strong architectural biases advantage compositional learning, highly engineered solutions prove difficult to compare rigorously against baselines. Ensuring fair comparisons requires investigating networks with strictly identical computational power, differing only in "push" toward structural modularity. We achieve this by optimizing very sparse networks under strict connectivity constraints modulated by spatial and neuromorphic priors.

The DEEP-R Algorithm

Training sparse neural networks without enforcing fixed initialization connectivity patterns remains challenging. We adopt the Deep Rewiring (DEEP-R) algorithm (Bellec et al., 2018), treating parameter space as a landscape for dynamic stochastic exploration. DEEP-R parametrizes each connection k with parameter θ_k and fixed sign $s_k \in \{-1, 1\}$ (determined at initialization). Effective weights w_k are defined via Rectified Linear Units (ReLU) on underlying parameters:

$w_k = s_k \cdot \max(0, \theta_k)$. Under this formulation, connections are *active* if $\theta_k > 0$ and *dormant* (logically disconnected) if $\theta_k \leqslant 0$.

Optimization updates active weights through three forces: gradient descent (standard back-propagation on task loss with learning rate η), regularization (L_1 penalty term $-\eta\alpha$ reducing weight magnitude to encourage sparsity), and stochastic exploration (random noise term $\sqrt{2\eta T}\nu$ where $\nu \sim \mathcal{N}(0, 1)$ and T is a temperature parameter, allowing random walks in parameter space). The update rule for active connections becomes:

$$\theta' \leftarrow \theta \; - \eta \frac{\partial L}{\partial \theta} - \eta\alpha + \sqrt{2\eta T}\nu \tag{2.1}$$

When active parameters θ_k cross zero and become negative, the connections deactivate. To maintain a strict connectivity budget, dormant connections are randomly selected and reactivated ($\theta_{k'} \leftarrow \epsilon$, ϵ small), ensuring constant network connectivity throughout training while the adjacency matrix evolves. Over the course of training, it is standard to decay the learning rate, to ensure that exploration rate drops accordingly, and encourage convergence to a final topology. Conversely, keeping the exploration term high at all time ensures the parameter space is continually explored, and endows the algorithm with continual learning abilities in the case of changing task distributions (Bellec et al., 2018).

Extending DEEP-R

We extend the original algorithm into a general framework designed for hardware-software co-optimization, defined by four sequentially applied components yielding final weight updates. The *base optimizer* is the (potentially stateful) mechanism transforming gradients into updates (e.g., SGD, Adam), which applied only to active weights. While this requires careful masking of 2nd order moments and momentums to active weights only (making sure deactivated weights stay truly frozen), the generalization of the algorithm to more powerful optimizers like Adam is beneficial. The *exploration strategy* is the stochastic process governing connectivity space search, also applied only to active weights, with exploration rates linked to current learning rates (enabling scheduling) and temperature parameter T. *Soft constraints* are differentiable priors acting on losses as regularization terms to "push" networks toward desirable topologies, ensuring hard constraint violations decrease as training advances. We apply all soft constraints to associated parameter groups at every step, only to active weights (this is the critical part that varies when training spatially embedded networks). Finally, *hard constraints* are non-differentiable physical constraints strictly enforced at every step (e.g., total synapse count, physical chip mapability, energy-usage), requiring manual verification of violations (e.g., active connection totals dropping below requirements), whereupon enforcement reoccurs (e.g., reactivating connections until reaching desired totals).

Spatial and Neuromorphic Priors

As primary application, we investigate sparse spatially embedded networks. Spatial constraints have been shownt to facilitate modularity and efficiency (Achterberg et al., 2023; Khona et al., 2023; Sheeran et al., 2024; Zhang et al., 2025). While imposing strict hard constraints on global connection numbers, we vary soft constraints (regularization penalties) based on hypothetical physical neuron embeddings. Penalties applied to sparse weights are computed from a penalty function that maps distances to precise values. These distances come from different spatial embeddings that provide a distance for any neuron pair (hence any weight in the parameters). Combining these two elements (a penalty function with a specific distance) yields penalty matrices for the networks' recurrent weight matrices. Effectively, we embed recurrent neurons in different topological spaces endowed with their own distances. This distance warping drives active weights into "cheaper" parameter space zones, converging toward local, low-distance connectivities (for the specific distance in effect). Note that, compared to other works like Zhang et al. (2025), we do not modulate weight reactivations based on distances (which is still performed uniformly). Deactivation pressure increases for long-range connections, resulting in more long-distance weight deactivation, though very strong and useful weights could remain, in typical brain economy trade-off fashion.

Baselines are non-spatial networks where L1 regularization applies uniformly across every connection, as is the case in the original DEEP-R work. We then define different spatial embeddings, which provides us with neuron distances (depicted in fig. 2.5). First, *geometric embeddings* use Euclidean distances $d(i, j)$ between Cartesian positions in space of neurons i and j. Placing neurons differently in space modulates precise distances (and thus penalties) associated with each recurrent matrix weight. We place neurons in Linear (1D line), Grid (2D plane), Cube (3D lattice), and Circular (circle circumference) configurations.

We introduce a specific *neuromorphic prior* based on the *Mosaic* architecture (Dalgaty et al., 2024): a memristor-based efficiency-designed architecture comprising Neuron Tiles (dense local connections) and Router Tiles (sparse global communication) (fig. 2.4). In this topology, connection costs are topological rather than Euclidean. We define the distance $h(i, j)$ as the integer number of router hops required to join the tiles of neurons i and j. To translate this discrete hop count into a regularization penalty, we define a target sparsity distribution $P(h) : \mathbb{N} \to [0, 1]$ (e.g., $\{0 : 1.0, 1 : 0.5, 3 : 0.1, \dots\}$), that specifies how sparse should neuronal communications be depending on their "hop" distance. The "effective distance" d_{eff} provided to the optimizer is derived from the inverse of this probability:

$$d_{\text{eff}}(i, j) \propto \frac{1}{P(h(i, j))}$$

This creates a warped metric space where neurons intended to be sparsely connected are mathematically "farther apart," forcing the optimizer to pay a higher penalty to maintain those

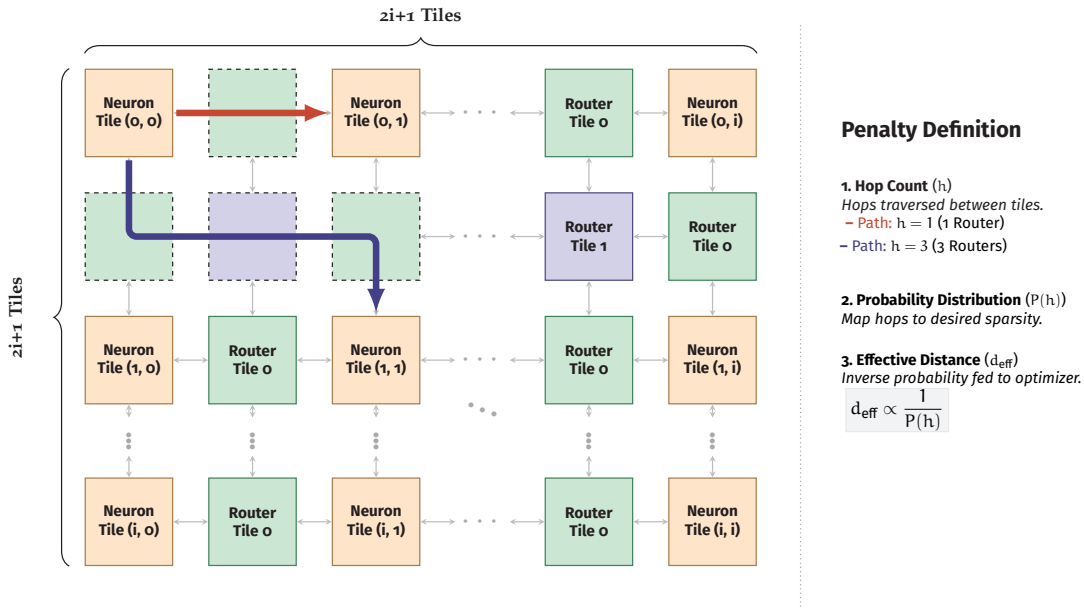


Figure 2.4: Illustration of the Mosaic neuromorphic architecture and penalty derivation. Neuron tiles (densely connected) are sparsely connected on a grid through router tiles. The diagram visualizes two connection lengths: a **red** 1-hop connection and a **blue** 3-hop connection. The panel on the right illustrates how the discrete hop count (h) is transformed into a target connection probability (P), the inverse of which serves as the effective distance (d_{eff}) for regularization.

connections. This approach grants fine-grained control over Mosaic connectivities, allowing us to experiment with many sparsity distributions, and steering networks toward efficient "small-world" topologies compatible with the hardware.

2.2.3 *Network Architecture*

We apply these constraints to Recurrent Neural Networks (RNNs). To process 2D grid columns, models are augmented with *Positional Attention Mechanisms*. This layer acts as the agent's "eye," translating internal pointer states into specific observations. Crucially, since pointers indicate *locations* (row indices) rather than content, we inject *learned positional encodings* into attention keys. At step t, query q_t is projected from previous hidden state h_{t-1}. Keys K_t are generated from current column C_t augmented with positional encodings P_{pos}, while values V_t remain raw column encodings:

$$q_t = W_Q h_{t-1} \tag{2.2}$$

$$K_t = W_K C_t + P_{pos} \tag{2.3}$$

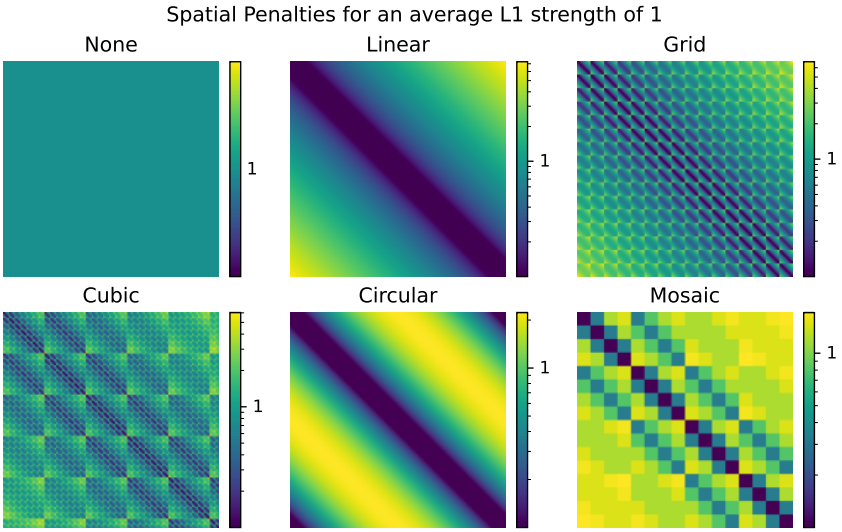


Figure 2.5: Regularization strength applied to each connection of a squared recurrent matrix of size 256, for different spatial embeddings of neurons. All weight matrices are regularized identically on average, but this regularization is distributed differently depending on the precise embedding. The Mosaic architecture used for this embedding is a 4x4 grid of neuron tiles, with a geometrically decreasing probability of connection per number of router hops.

Attention scores are computed via scaled dot-product $\alpha_t = \text{softmax}(\frac{q_t K_t^T}{\sqrt{d}})$. Final inputs to RNNs are weighted sums of the cells in the original column $x_t = \sum_i \alpha_{t,i} C_{t,i}$. Attended inputs x_t are then passed to RNN cells:

$$h_t = \tanh(W_{in} x_t + W_{rec} h_{t-1} + b) \tag{2.4}$$

Spatial embeddings described above are applied specifically to recurrent matrices W_{rec}. Networks have two output heads: a *Transformation Head* (attached) project h_t to predict target state y_t (the main training objective), and a *Pointer Head* (detached) used as linear probe attempting to decode current target row indices from h_t. Gradients from the pointer head are not back-propagated back to RNNs, allowing monitoring of whether models "solved" symbolic pathfinding without leaking this information to main models (this is only used as a tool for us to understand networks, and is not part of task-solving).

2.2.4 *Statistical Framework: Matched-Pairs Analysis*

Training sparse recurrent networks involves significant stochasticity, where initialization seeds, and random data seeds, can drive large variance in final performance. To isolate the causal impact of spatial priors from this background noise, we employ a rigorous **matched-pairs design**.

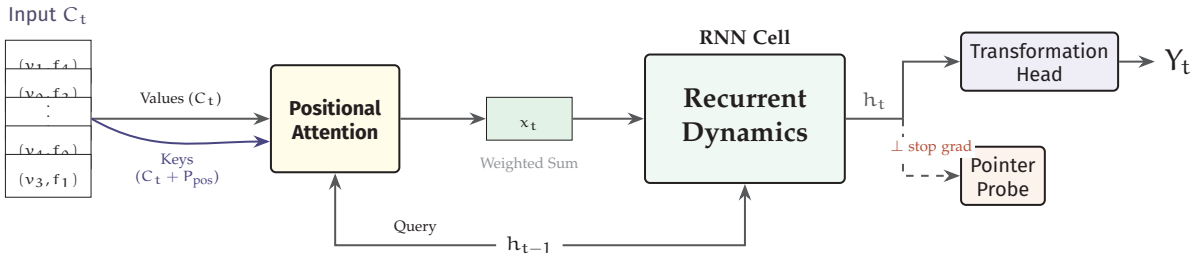


Figure 2.6: Summary Figure of the architecture and data flow

Paired Experimental Design

The goal here is to compare spatially-embedded networks to baselines, "everything else being equal". Rather than comparing population averages, we pair every spatially embedded network (treatment) with a non-spatial network (control) that is identical in all other factors (sharing the same random seed, function library, width, and depth). By analysing the *distribution of differences* within these pairs, we effectively subtract out the variance attributable to initialization in the model or variations in the dataset, allowing us to detect effects that might otherwise be masked by the high variance of sparse training.

Fair Comparisons via Max-Performance Aggregation

A critical challenge is that spatial and non-spatial networks may require different regularization strengths to reach optimal performance. Comparing them at a fixed regularization coefficient would favour whichever model happens to be better aligned with that specific value. To ensure a fair comparison, we adopt a **max-performance** protocol. For each unique experimental configuration (seed and dataset), we sweep over a range of regularization coefficients (λ) for both the control and treatment groups. We then select the single best-performing model (highest validation accuracy) from the control sweep and pair it with the best-performing model from the treatment sweep. This ensures that we are comparing the spatial prior at its best against the baseline at its best, eliminating hyperparameter sensitivity as a confounding variable.

Hypothesis Testing

To determine if the observed differences are statistically significant, we use **two-sided** hypothesis tests. This means we do not assume in advance that spatial networks will be better; we simply test if they are *different* from the baseline. We use the **Wilcoxon Signed-Rank Test**. The test ranks the magnitude of the differences and checks if the positive differences consistently outweigh the negative ones (or vice-versa). We consider a result significant if the p-value for

the test is less than 0.05. In every figure comparing treatments to baselines, comparisons are plotted in **green** when *statistically significantly higher*, **red** when *statistically significantly less*, and **grey** when the differences are *not statistically significant*.

2.2.5 *Functional Analysis: The Selectivity-Clustering-Lesion Pipeline*

To move beyond aggregate performance metrics and rigorously test the "Structure ↔ Function" hypothesis, we developed a three-stage analysis pipeline. This framework operates on the premise that "aligned" modularity implies a causal alignment between physical clusters (structural modules) and task-specific neuron groups (functional modules). All analyses are performed on the Out-of-Distribution (OOD) dataset split to capture robust generalization properties.

Stage 1: Context-Based Selectivity Analysis

First, we characterize the *potential* functional role of each neuron by measuring its responsiveness to specific task contexts. Inspired by predictive coding analyses (Boeshertz and Clopath, 2025) and task-selectivity metrics (Yang, 2019), we define N_{ctx} discrete contexts based on the active function at each timestep (e.g., C_k = "Time steps where function f_k is applied").

We collect neuron activations across the dataset and compute the mean activation $\mu_{i,k}$ for each neuron i within context k. To filter out task-irrelevant units, we exclude neurons where the variance of these context means falls below a threshold $\epsilon = 0.01$. For the remaining neurons, we compute a selectivity profile vector $\mathbf{s}_i \in \mathbb{R}^{N_{ctx}}$. We quantify selectivity $s_{i,k}$ as the normalized squared deviation of the context mean from the global mean $\bar{\mu}_i$:

$$s_{i,k} = \frac{(\mu_{i,k} - \bar{\mu}_i)^2}{\max_j (\mu_{i,j} - \bar{\mu}_i)^2} \tag{2.5}$$

This normalization scales the profile to $[0, 1]$, ensuring that clustering is based on the relative preference structure of the neuron rather than its absolute activation magnitude.

Stage 2: Dual Clustering Tracks

To test for alignment between topology and function, we partition the network's hidden layer using two independent clustering modalities:

1. Structural Clustering (The "Hardware" View): We treat the recurrent weight matrix W_{rec} as the adjacency matrix of a weighted graph. We apply the **Louvain community detection**

algorithm (Blondel et al., 2008) to maximize graph modularity Q, identifying groups of neurons that are densely connected to each other but sparsely connected to the rest of the network.

2. Functional Clustering (The "Software" View): We cluster neurons solely based on their selectivity profiles $\mathbf{s}_i$. We apply **Ward hierarchical clustering** (Pedregosa et al., 2011) to these feature vectors, automatically selecting the optimal number of clusters $k \in [2, 75]$ that maximizes clustering score. This groups neurons that "care" about the same sub-functions, independent of their physical location in the network.

Stage 3: Causal Lesioning and Specialization Scores

Correlation does not imply causation; a neuron may be selective for a function without being necessary for its execution. To measure necessity, we perform **Instantaneous Causal Lesioning**.

For a given cluster $\mathcal{C}$ (whether structural or functional) and timestep t, we ablate $\mathcal{C}$ by forcing its activations to the empirical cluster mean *only* at step t. This is performed over 10 data batches ($\approx$ 2560 samples). We measure the relative performance drop on specific contexts, yielding an Ablation Matrix $\mathbf{A} \in \mathbb{R}^{N_{ctx} \times N_{clusters}}$, where A_{kj} represents the normalized impact of ablating cluster j on context k (0 implies no impact, 1 implies a drop to chance accuracy).

Finally, we quantify specialization using information-theoretic entropy. Crucially, we define our metrics using **entropy scores**, such that higher values indicate higher specialization (lower entropy).

The **Module Specialization Score** S_{mod} for a cluster j is defined as:

$$S_{mod}(j) = 1 - \frac{H(\mathbf{p}_j)}{\log(N_{ctx})} \tag{2.6}$$

where H denotes the Shannon entropy and $\mathbf{p}_j$ is the impact distribution of cluster j across all contexts. A score $S_{mod} \approx 1$ implies the cluster is a "specialist" (highly impacting only one specific function), while $S_{mod} \approx 0$ implies a "generalist" (impacting all functions equally).

Complementarily, we define the **Task Localization Score** S_{task} for a sub-routine i as:

$$S_{task}(i) = 1 - \frac{H(\mathbf{p}_i)}{\log(N_{cluster})} \tag{2.7}$$

where $\mathbf{p}_i$ is the impact distribution on task i from all clusters. This metric quantifies how "localized" a task is; a high score indicates the task relies on a single cluster, whereas a low score suggests the computation is distributed across the network.

2.3 RESULTS

We present systematic analysis of spatially embedded sparse recurrent networks trained on the 2D-PVR meta-dataset, investigating whether spatial embedding serves as beneficial inductive bias for compositional reasoning and generalization. To ensure rigorous conclusions, we employ matched pair hypothesis testing frameworks: networks with and without spatial embeddings train across wide sweeps of model and data parameters, testing for significant performance differences between embedded treatments and null baselines (no spatial constraints) while controlling for all other factors (see section 2.2.4). This effectively answers: *everything else being equal, are spatial embeddings a* ***net*** *advantage for sparse networks tackling compositional tasks*?

2.3.1 *Sanity Check: Static vs Dynamic optimization*

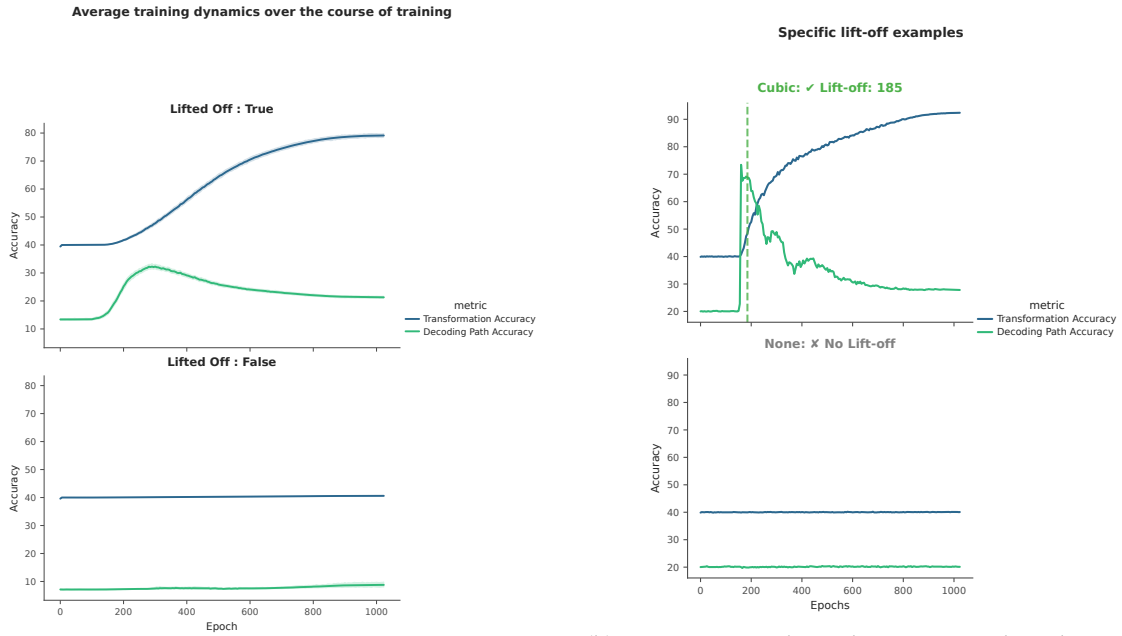
As a sanity check to confirm the effectiveness of our optimization method (see section 2.2.2), we compare our *dynamic baseline* (uniform L1 regularization) against sparse, *static* networks. We find that using our deep-R dynamic rewiring improves performance dramatically (see fig. B.1), and we therefore only focus on dynamically rewired networks from this point forward, investigating the impact of spatial embeddings as our main treatment of interest.

2.3.2 *Task Difficulty and Learning Dynamics*

We first characterize the proposed 2D-PVR task's difficulty. While networks successfully learn to chain transformations through small grids, performance degrades steeply as grid size increases (fig. B.1). Our main performance metric is length generalization: while networks train on 10-timestep sequences, we evaluate them on length-20 sequences: this is our OOD accuracy metric. Interestingly, we observe "grokking"-like behaviour where OOD accuracy remains at chance level for extended periods before exhibiting phase transitions where accuracy takes off (fig. 2.7b, top row). Crucially, significant fractions of models and some hyperparameters fail to "lift off" entirely, remaining stuck in initial plateaus.

To peer into hidden learning dynamics, we trained secondary, detached linear readouts on RNN frozen hidden states to predict current timestep path locations (thus not interfering with training nor providing information to main learning models, see section 2.2.3). Since correct pointer inference is a prerequisite for selecting correct functions, this serves as a proxy for models' intermediate understanding. In successful runs, we observe that pointer decoding accuracy rises sharply *before* transformation accuracy begins ramping up (see the general trend in fig. 2.7a as well as a specific example in fig. 2.7b, top row) suggesting that inferring control flow is a necessary first curriculum step. Interestingly, decoding performance then decays on

average after task mastery, suggesting RNN representations of pointers evolve into formats less accessible to linear probes, or that control flow integrates more "holistically" into computation. In failed runs, path-decoding accuracy fails to increase entirely.



(a) Average training dynamics, for all models, depending on whether the run "lifted-off" or not.

(b) Training example, with one run stuck at chance while another starts to generalize around epoch 185

Figure 2.7: Training and "lift-off" dynamics

2.3.3 *Enhanced Generalization Under Spatial Constraints*

We then investigate generalization ability of trained networks more closely. Crucially, both *depth generalization* (applying learned algorithms to sequences longer than training-observed, probing whether models learned symbolic algorithms rather than memorizing fixed-length patterns) and *systematic generalization* (handling novel sub-function combinations or path structures absent from training sets) can be seen as included in the longer OOD examples, since these contain novel combinations of functions from unseen navigational paths, and longer examples than those of the training set.

First, spatially embedded sparse networks display significantly higher overall generalization performance (fig. B.1). However, stratified analysis based on regularization strength (λ) reveals nuance. Spatial embeddings seem most effective at higher regularization strengths where connection pruning pressure is strong (fig. B.2 lower left panel), while baselines seem to perform best with lower regularizations. At lower regularization strengths, spatial embeddings actually can be neutral or slightly detrimental compared to baselines (fig. B.2 top row). This

suggests optimal regularization strength differs for baselines and treatments and should not be treated as mere additional parameters.

To account for this interaction, we remove L1 strength from paired parameters and instead aggregate results using *max-performance* protocols: for every data configuration, random seed, and spatial embedding, we compare *best performing* spatial models (sweeping over λ) against *best performing* baseline models (see section 2.2.4 for complete details of our statistical analysis). Even under this stricter comparison, spatially embedded networks significantly outperform baselines on generalization accuracy (fig. 2.8, top left). This indicates advantages are not hyperparameter tuning artifacts but robust properties of inductive biases, guiding networks toward topologies inaccessible to baselines.

Additionally, given "grokking" dynamics described previously, we define two "lift-off" metrics: a boolean indicating whether the run ever left the initial chance plateau (OOD accuracy consistently exceeding chance by 5%), and the epoch at which it did (if ever). We empirically find this marks transitions from flat loss plateaus to active learning phases (though not all runs reach subsequent perfect performance). Spatially embedded networks exhibit both significantly more runs which enter this generalization period (lift-off proportion) and display earlier lift-off times compared to null baselines (fig. 2.8 top right and bottom left panels). This suggests that inductive biases help models navigate the initial deceptive landscape phase better, and seem to provide a substantial *sample efficiency* improvement.

2.3.4 *Structural Modularity Emergence*

We analysed resulting trained network topologies using modularity Q-metrics (Newman, 2006) on both binary adjacency matrices and weighted connectivity matrices. Dynamic rewiring of sparse networks successfully guides recurrent weights toward locally clustered graphs, as can be seen in one example in fig. 2.9. We report several findings. First, pure L1 regularization alone induces some degree of modularity in effective weights (Q measured on weighted matrices rises during training), even without spatial constraints, confirming prior observations that sparsity training alone can induce modularity (see fig. B.3, top-left panel). Nonetheless, spatially embedded networks achieve significantly higher structural modularity than baselines (fig. 2.8 bottom right panel). This is true even while only comparing *best-performing* models from each group since, as discussed, high regularization works overall better for spatial embeddings, subsequently leading to more structurally modular topologies.

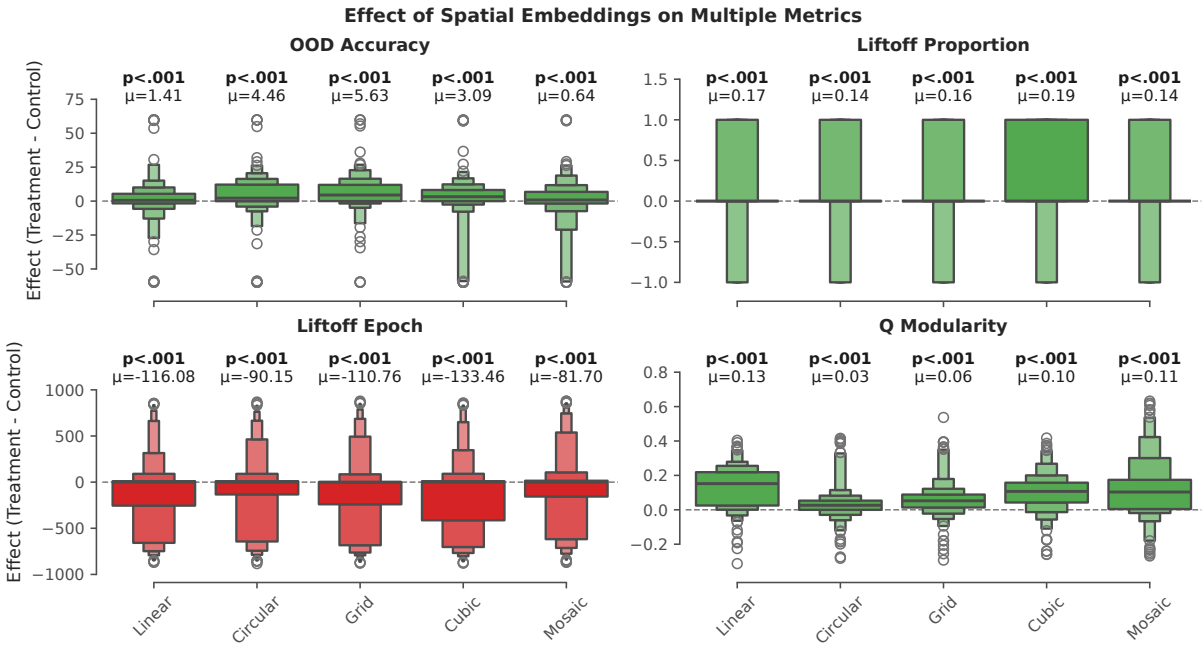


Figure 2.8: Statistical Analysis of the effect of treatments (spatial embeddings) vs the control baseline (uniform regularization), for different metrics. For every plot, we display p-values (bold if significant), average difference μ, and a boxen plot of the differences from matched pairs. Each box is coloured *green* when statistically significantly higher, *red* when statistically significantly less, and *grey* when the differences are non-significant. Metrics presented are : **a**./ Out of distribution accuracy, **b**./ boolean "lift-off" indicator showing initial random-chance plateau was left, **c**./ "lift-off" epoch, denoting at which epoch generalization started (if it ever did) and **d**./ Q metric modularity on the final weight matrices

2.3.5 *Width-Dependent Effects*

We further stratified analysis by *compositional width* (function library size) of datasets. Counter-intuitively, lower compositional widths lead to higher variance in final training performance. We hypothesize that when function libraries are small, datasets contain fewer distinct instruction patterns, making it harder for models to disentangle general algorithms from specific data regularities. Conversely, high-width settings provide richer signals for learning abstract composition and pathfinding rules. Nonetheless, spatial embeddings are significantly more effective in high-width regimes (fig. 2.10). This could align with hypotheses that modular architectures (encouraged by spatial constraints) are particularly beneficial for highly compositional problems where distinct sub-routines must be isolated and recombined. We can in fact notice a higher rise of the structural modularity at higher compositional width during training, as seen in fig. B.3. We should nonetheless be wary of not confounding *correlation* with *causation*, and structural modularity and improved generalization at high-width could be two co-occurent effect of the same underlying cause.

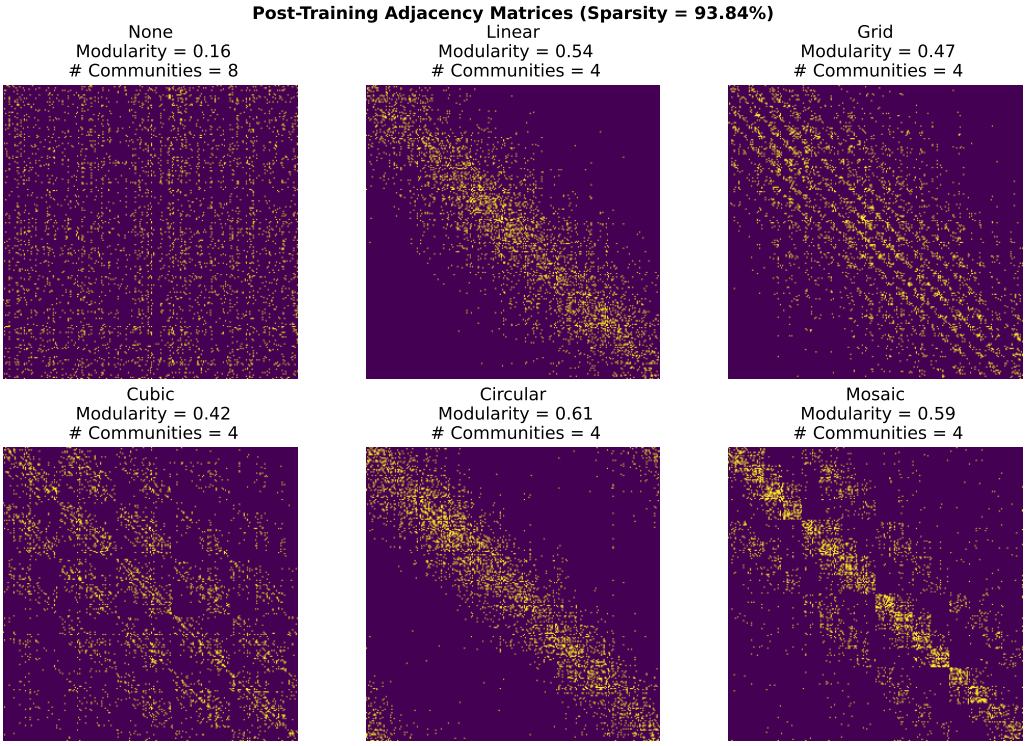


Figure 2.9: Example of final adjacency matrices (of the recurrent weights) for multiple spatial embedding, alongside the baseline, trained from the same initialization, under the exact same strict connection count.

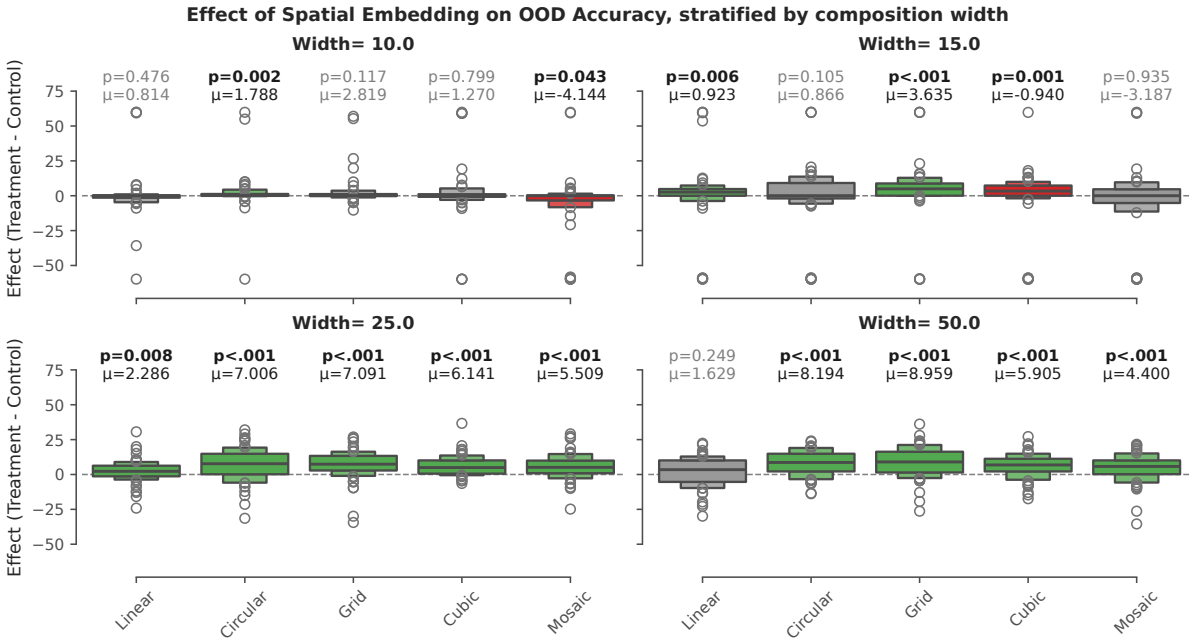


Figure 2.10: Effect of spatial embeddings on generalization accuracy, stratified by compositional width.

2.3.6 *The Structure-Function Gap: Modularity without Specialization*

A central hypothesis driving this investigation—and indeed, much of the Brain Economy literature—is that spatial constraints enforce structural modularity, which in turn fosters functional specialization. The logic follows that by penalizing long-range communication, the network is forced to colocate neurons processing related sub-tasks, leading to the emergence of distinct functional modules. This hypothesis was recently strengthened by several findings in the field (Achterberg et al., 2023; Zhang et al., 2025). To test this, we applied the Selectivity-Clustering-Lesion pipeline (defined in section 2.2.5) to our trained networks. If the hypothesis holds, spatially embedded networks should exhibit significantly higher **Module Specialization Scores** (clusters act as specialists) and **Task Localization Scores** (sub-routines rely on specific clusters) compared to their non-spatial baselines. A cherry-picked example of the functional analysis conducted on a single trained network is shown in fig. B.4.

Contrary to our expectations, we observe that spatial constraints yield only *very marginal* differences in functional specialization. As illustrated in fig. 2.11 (top row), when we analyse the **Functional Track** (where neurons are clustered strictly by their selectivity profiles), the shift in specialization scores between spatial models and their matched baselines is negligible. While some embeddings show statistically significant improvements ($p < 0.05$), the *magnitude* of these effects (μ) is in the order of 10^{-3} (on a normalized scale of $[0, 1]$). In many cases, the difference is statistically indistinguishable from the baseline or even detrimental. This indicates that while spatial networks are indeed more structurally distinct, their neurons do not become significantly more "opinionated" or specialized regarding specific atomic functions than neurons in our baseline sparse network. The disconnect becomes even more apparent when analysing the **Structural Track** (fig. 2.11, bottom row). Here, clusters are defined by the physical connectivity (Louvain communities). We previously established that spatial embeddings significantly increase the modularity Q of these physical weights (see fig. 2.8, panel d). However, our causal lesioning results reveal that these physical modules do not map significantly better onto functional tasks in spatial networks.

These findings present an interesting paradox. We have established that spatial embeddings significantly improve OOD generalization and learning efficiency (section 2.3), yet they do not appear to do so by creating "clean," interpretable functional modules. This stands in contrast to the recent observations of Zhang et al. (2025), who, using metrics analogous to ours (module and task variance), reported significant structure-function alignment in spatially constrained networks. A likely differentiator is the nature of the task: our 2D-PVR domain requires dynamic, algorithmic execution where control flow (pointers) and data transformation (instructions) are inextricably linked. These results are also likely to reflect the "entanglement" phenomena observed in Chapter 1. It appears that while spatial priors does force the network to organize into local clusters to save energy (minimizing wiring cost), the *computation* remains distributed.

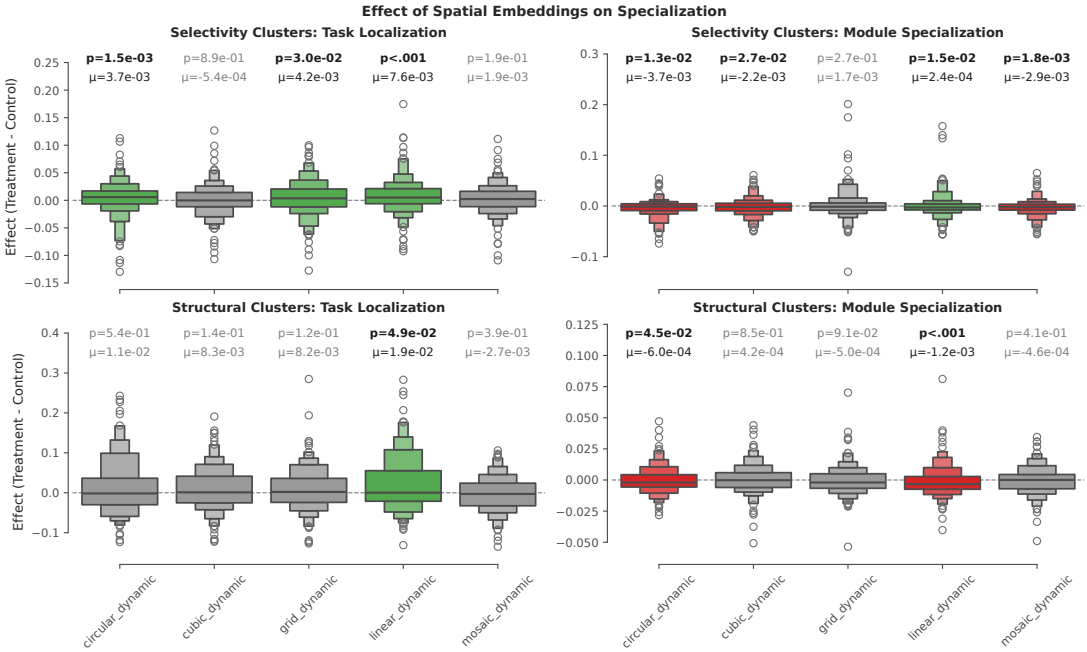


Figure 2.11: Specialization Results: Functional and Structural Task and Module Entropy

Consequently, we may require additional constraints—beyond simple distance penalties—to force the emergence of meaningful functional specialization.

2.4 CONCLUSION

We introduced a dual framework designed for the mechanistic investigation of compositional learning under spatial constraints. The *2D-PVR meta-dataset* serves as a tunable testbed, enabling the rigorous disentanglement of mutiple axes of complexity from distributional shifts. Complementing this, the *Spatial DEEP-R* framework provides a hardware-software co-design tool capable of injecting biologically inspired geometric priors into sparse recurrent networks while maintaining strict computational equivalence with baselines.

Our results demonstrate that spatial embeddings constitute powerful inductive biases for compositional reasoning. By penalizing long-distance connections, these priors significantly enhance both systematic generalization and learning efficiency, enabling networks to "lift off" from chance-level plateaus earlier than their non-spatial counterparts. Notably, these advantages are most pronounced in high-width compositional regimes, suggesting that spatial constraints are particularly beneficial when agents must navigate large libraries of atomic primitives. However, the mechanism driving this performance proves nuanced. While spatial embeddings reliably induce high structural modularity (clustering of physical weights), this does not translate into a simple one-to-one correspondence with functional specialization. The

disconnect between clean topological modules and distributed functional execution suggests that spatial priors may drive generalization through less intuitive mechanisms.

Our evaluation of the specific *Mosaic* neuromorphic prior offers a promising validation for hardware co-design. Crucially, the prior successfully induced hardware-compliant small-world topologies while still delivering small, but significant performance gains over non-spatial baselines. Although generic, relaxed geometric embeddings (such as simple grids) seem to achieve slightly faster convergence and better generalization, the Mosaic prior demonstrates that strict hardware constraints—such as those required by physical memristor arrays—need not be a barrier to effective learning. Instead, we show that it is possible to respect the rigorous energy-efficiency requirements of physical substrates while simultaneously leveraging spatial inductive biases to improve algorithmic performance.

The potential of the 2D-PVR framework remains largely untapped. While this work focused on the "direct" composition of functions, future iterations should exploit the framework's flexibility to explore the tension between perceptual learning and symbolic reasoning. Two specific avenues appear most promising. First, increasing navigational complexity—by introducing relative addressing or complex pointer logic—could force the emergence of deeper algorithmic reasoning beyond simple indexing, and introduce longer time-dependencies. Second, we envision extending the dataset to include complex *perceptual encodings*. By replacing symbolic inputs with high-dimensional embeddings or images, we can force the network to develop "decoder" modules alongside its reasoning core. Investigating how spatial constraints mediate the competition between learning these perceptual sub-routines and maintaining symbolic control flow offers a fertile ground for understanding the complexity of true neuro-symbolic architectures.

Ultimately, by bridging neuroscience-inspired principles with systematic benchmarking, this framework opens new avenues for understanding compositional intelligence in artificial systems.

Part II

SELF-ORGANISING PRINCIPLES OF INTELLIGENCE

3

A PATH TO UNIVERSAL NEURAL CELLULAR AUTOMATA

Narrative Preamble

The first half of this thesis investigated modularity through the lens of architectures and constraints. In the second half, we step back from cleanly defined models, with specific priors, to explore a more fundamental aspect of intelligence: *Self-Organization*. This chapter represents a theoretical deep dive into the capabilities of Neural Cellular Automata (NCA). While typically studied in the context of morphogenesis (growing static patterns like images) we ask here whether these systems can serve as a substrate for general-purpose computation. This is an exploration of the low-level principles required to "compile" functional tasks into a continuous, locally-connected medium. We do not yet ask how a brain grows, but rather: can a self-organizing system of local rules perform the precise, universal computations (like matrix operations) that we usually reserve for engineered silicon?

Publishing Context

This work was simultaneously published in the 2025 "GECCO" conference (**Gabriel Béna** et al., 2025b) and as a web-version "distill-style" article (in true NCA research fashion).

Contributions

The author and Maxence Faldor contributed equally to this work. Maxence Faldor initiated the project, conceived its central ideas, including the concept of state and immutable state for hardware representation. He also developed the initial software library and experimental design. The author provided crucial feedback on these foundational ideas, developed the tasks setup, led the development of the compiler and the Graph Neural Network (GNN) components, conducted the extensive experiments, and was the primary author of the paper and website. Dan Goodman and Antoine Cully supervised the project, offering guidance, insightful discussions, and contributing to brainstorming sessions.

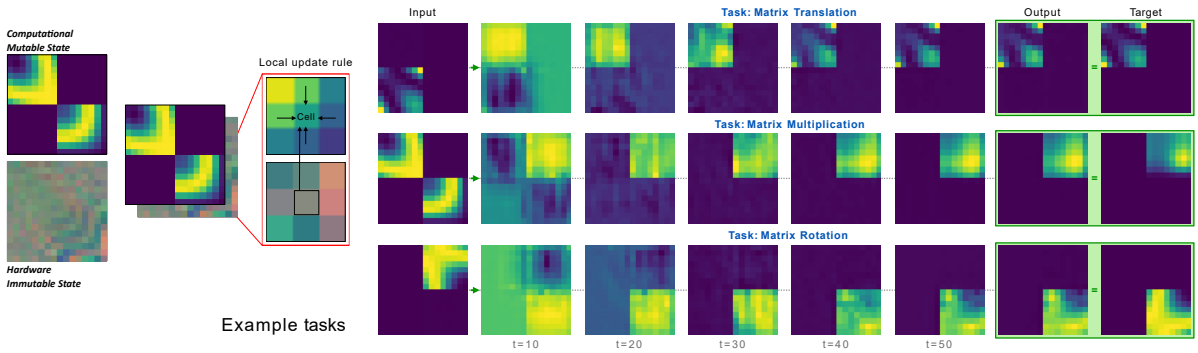


Figure 3.1: Our framework performs matrix operations (translation, multiplication, rotation) through continuous cellular interactions evolving over time (from $t = 0$ to $t = 60$). The inset reveals the core mechanism: cells communicating only with neighbors, and interacting with a local fixed (but learned) heterogeneous substrate, are able to collectively solve mathematical tasks without explicit algorithms.

3.1 ABSTRACT

Cellular automata have long been celebrated for their ability to generate complex behaviours from simple, local rules, with well-known discrete models like Conway's Game of Life proven capable of universal computation. Recent advancements have extended cellular automata into continuous domains, raising the question of whether these systems retain the capacity for universal computation. In parallel, neural cellular automata have emerged as a powerful paradigm where rules are learned via gradient descent rather than manually designed. This work explores the potential of neural cellular automata to develop a continuous Universal Cellular Automaton through training by gradient descent. We introduce a cellular automaton model, objective functions and training strategies to guide neural cellular automata toward universal computation in a continuous setting. Our experiments demonstrate the successful training of fundamental computational primitives — such as matrix multiplication and transposition — culminating in the emulation of a neural network solving the MNIST digit classification task directly within the cellular automata state. These results represent a foundational step toward realizing analog general-purpose computers, with implications for understanding universal computation in continuous dynamics and advancing the automated discovery of complex cellular automata behaviours via machine learning.

3.2 INTRODUCTION

Cellular Automata (CA) represent a fascinating class of computational models that have captivated researchers across disciplines for their ability to produce complex behaviours from simple rules (Flake, 1998). At their core, CA typically consist of a grid of cells, each in one of a finite number of states, which evolve over discrete time steps based on a fixed deterministic

rule. This rule, applied uniformly to all cells, governs the state transition of each cell based solely on its current state and those of its neighbors. Despite their simplicity, CA have become a cornerstone in studying emergence and complexity (Wolfram, 1984). Mathematical proofs have established that well-known CA, such as Conway's Game of Life, Elementary Cellular Automata, and Wireworld, are capable of universal computation, underscoring their remarkable power and expressiveness (Cook, 2004; Rendell, 2016). Beyond these formal demonstrations, researchers have constructed fully functional Turing machines within these CA, albeit through arduous efforts requiring meticulous design and substantial time investment.

In recent years, the development of continuous CA has enabled to bridge the gap between the discrete nature of traditional models like Conway's Game of Life and the analog characteristics of the real world. Notable examples include Lenia (Chan, 2019) and SmoothLife (Rafler, 2011), which extend classic CA to simulate lifelike patterns with continuous dynamics. However, a key open question persists: are these models capable of universal computation? While the answer is likely affirmative, given their expressive potential, proving this remains elusive. The lack of discrete states and well-defined transitions makes it harder to encode symbolic information reliably — slight perturbations can lead to significant divergence over time, undermining the stability required for computation. Moreover, in contrast to the sharp boundaries and interactions of traditional CA, continuous models often exhibit smooth, fuzzy dynamics that make it challenging to design modular components like wires, gates, or memory elements with predictable behavior. What was already a laborious task in discrete CA becomes more difficult, if not practically impossible, in the continuous domain, highlighting a fundamental challenge in building efficient analog computers.

In parallel with these developments, Neural Cellular Automata (NCA) have emerged as a compelling paradigm that blends the local, decentralized dynamics of classical CA with the representational capacity and trainability of neural networks (Mordvintsev et al., 2020). Unlike traditional CA, where the update rule is explicitly handcrafted, NCA leverage differentiable architectures wherein the rule is parameterized by a neural network and optimized end-to-end via gradient descent. This makes it possible to *learn* complex behaviours and dynamics from data, bypassing the need for manual rule design. Recent work has demonstrated that NCA can be trained to perform a wide range of tasks: from self-organizing into complex morphologies (Mordvintsev et al., 2020), to solving algorithmic reasoning tasks such as instances of the 1D-ARC challenge (Faldor and Cully, 2025), exhibiting emergent collective behaviours (Randazzo et al., 2020), and growing artificial neural networks (Najarro et al., 2022*b*). These results highlight the versatility of NCA as a model of computation and pattern formation. Given the Turing completeness of classical CA, NCA offer an exciting new lens through which to explore the space of rules — not by manually engineering them, but by searching for them through optimization. In essence, NCA turn rule discovery into a machine learning problem. This shift is significant: the traditionally arduous task of hand-crafting rule sets that give rise to desired behaviours is now offloaded to the learning algorithm itself. Backpropagation through

time, combined with the differentiable nature of NCA, allows for flexible, automated tuning of highly non-trivial dynamics.

In this work, we explore the potential of the Neural Cellular Automata paradigm to pioneer the development of a continuous Universal Cellular Automata (Weisstein, n.d.), with the ambitious goal of inducing a universal Turing machine (Turing, 1937) to emerge within a continuous CA through training via gradient descent. The development of Universal Neural Cellular Automata has implications beyond academic curiosity or creating interesting simulations. It touches on fundamental questions about the potential for continuous dynamic systems to exhibit universal computation, or the possibility to create a universal analog computer. This paper establishes the foundational framework and presents promising initial steps toward realizing this grand vision. First, we propose a novel framework that disentangles the concepts of "hardware" and "state" within NCA. In this abstraction, CA rules serve as the "physics" dictating state transitions across space and time, akin to the fundamental laws governing computation. The CA state, in turn, acts as the dynamic physical substrate — comparable to electrical charges in a computer or neurochemical patterns in a brain—while the "hardware" represents an immutable scaffold, fixed in its spatial configuration throughout a simulation. This hardware can be leveraged by the physics (i.e., the CA rules) to guide computation but remains unalterable during runtime, providing a stable backbone for emergent behaviours. Second, we introduce preliminary objective functions and training setups designed to steer NCA toward universal computation in a continuous domain. Third, we conduct experiments demonstrating the training of essential computational building blocks — such as matrix multiplication, dot-product and transposition — within the NCA framework. Finally, we showcase the practical utility of these building blocks by emulating a neural network directly within the mutable CA state, successfully solving the MNIST digit classification task. These results mark a critical first step, illustrating how NCA can harness gradient descent to sculpt continuous CA into powerful, general-purpose computational systems.

3.3 RELATED WORK

Discrete CA have long been a cornerstone for studying universal computation due to their ability to generate complex behaviours from simple, local rules. A number of discrete CA, such as Conway's Game of Life (Gardner, 1970; Rendell, 2016), Rule 110 of Elementary Cellular Automata (Cook, 2004; Wolfram, 2002), Langton's Ant (Gajardo et al., 2002), and Wireworld (ADA University et al., 2018) have all been proven Turing-complete. These works rely on discrete states and labor-intensive, hand-crafted designs. To mitigate this, evolutionary algorithms have been employed to automate the discovery of CA rules or patterns with specific properties (Mitchell et al., 2000; Sapin et al., 2003), reducing human effort though still targeting discrete systems and predefined goals rather than general-purpose computation.

The shift to continuous CA aims to connect discrete models with real-world analog systems, prompting inquiries into their computational potential. Models like Lenia (Chan, 2019) and SmoothLife (Rafler, 2011) introduce smooth state transitions, yielding lifelike emergent patterns, yet their capacity for universal computation remains unproven. Recent efforts have applied evolutionary search to Lenia (Faldor and Cully, 2024; Reinke et al., 2020) to automatically discover and optimize patterns, though these pursuits prioritize specific behaviours over general-purpose computation. Similarly, gradient descent has been used on Lenia patterns and rules to discover and optimize patterns (Hamon et al., 2024), though again not targeting general-purpose computation.

NCA mark a paradigm shift by replacing hand-crafted rules with neural networks trained via gradient descent. NCA have been successfully applied to specific tasks such as morphogenesis (Mordvintsev et al., 2020), classification (Randazzo et al., 2020), and solving difficult problems like the 1D-ARC challenge (Faldor and Cully, 2025). Most relevant to our work, HyperNCA (Najarro et al., 2022*b*) uses NCA to grow artificial neural networks, suggesting broader computational versatility. These advances highlight NCA's strength in automating rule discovery, offloading the burden of manual design to machine learning. Nevertheless, prior NCA research predominantly focuses on pattern formation or specific tasks. It is to be noted that the matrix copy and multiplication tasks were first implemented with NCA by Peter Whidden (Whidden, 2020), and that our approach builds upon this idea that NCA can be used in a continuous computational setting. Very recently, (Miotti, Pietro and Niklasson, Eyvind and Randazzo, Ettore and Mordvintsev, Alexander, 2025) have also demonstrated that NCA rules can be implemented using (differentiable) logic gates, confirming the possibility to run self-organizing systems on standard digital harware. Finally, in parallel of our work (Guichard et al., 2025) was developped using a similar idea: cells augmented with "private memory tapes" were shown to exhibit stable, multi-task capabilities.

Our research intersects with analog computing and particularly neuromorphic approaches that bridge biology and silicon. Analog computation utilizes continuous physical systems, and leverages physical phenomena such as wave propagation, diffusion, and material properties to represent and transform information continuously, avoiding the quantization overhead of digital systems (Ulmann, 2022). NCA draw inspiration from biological neural networks, which primarily employ local computations for energy efficiency (Bassett and Bullmore, 2006; Bullmore and Sporns, 2012*b*; Meunier et al., 2010). This principle of locality is fundamental to both systems, and demonstrates that sophisticated computation can emerge from simple, localized rules without requiring global connectivity. The brain's co-location of computation and memory offers a solution to the von Neumann bottleneck that increasingly limits conventional computing systems as model complexity grows (Backus, 2007). Neuromorphic systems implement this biological principle through distributed processing elements with local memory, often using mixed-signal circuits that approximate neural dynamics while maintaining energy efficiency (Christensen et al., 2021; Mead, 1990; Schuman et al., 2017). While deep learning

has flourished through hardware-software co-design optimized for parallel matrix operations, this specialization has simultaneously restricted algorithmic innovation to operations aligned with current hardware capabilities. By exploring how systems with local interactions like NCA can implement universal computation, we can develop more versatile computing architectures that maintain the locality constraints of biological systems while leveraging silicon's speed advantages. This approach may yield computing systems that better balance computational power with the remarkable efficiency and adaptability characteristic of biological intelligence.

3.4 METHODS

We leverage the CAX (Faldor and Cully, 2025) library for high-performance neural cellular automata implementions and run our experiments on a single L40 GPU.

3.4.1 *General Setup*

The goal of our framework is to demonstrate the ability of neural cellular automata to act as a general computational substrate. To do so, we tackle a variety of tasks, directly in the NCA state (see 3.4.4). In this work, we introduce a novel architecture design that enhances such computational capabilities. Our approach partitions the NCA state space into two distinct components.

- The mutable state: this serves as the main workspace (where the tasks inputs are transformed into their outputs). This state is the only one changing through time during a single experiment / task roll-out. This is the computational substrate, meaning that tasks (such as matrix operations) are directly embedded in this space, and transformations on inputs have to happen in this state. The update dynamics of the mutable state are governed by the NCA rules, described in section 3.4.2.
- The immutable state: this functions as a specialized hardware configuration which is spatially heterogeneous. This hardware can itself be monolithic (the same shape as the entire grid) or modular (created from different specialized components for each task specific instance). This is learned across training but fixed in any duration of a single experiment / task instance. Details are specified in section 3.4.3

Overall, this framework enables a two-level optimization strategy: At the global level, we train a general-purpose NCA rule (perceive and update functions) to support diverse computational operations. At the task-specific level, we optimize individual hardware configurations. The system achieves task-specific computation by adapting its dynamics using the local available hardware (reminiscent of placing the correct components on a motherboard). From an efficiency

perspective, this architecture also provides significant practical advantages: once the general NCA rule is trained, adapting the system to new tasks requires only the optimization of hardware configurations, a process that is substantially less computationally intensive than training the full NCA rule from scratch. Full training details are described in section 3.4.5.

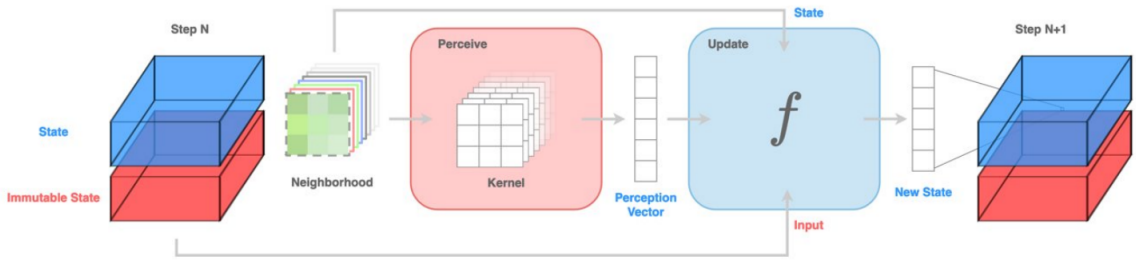


Figure 3.2: Schematic of our architecture, showing the distinction between mutable (computational) and immutable (hardware) states.

3.4.2 *Neural Cellular Automata*

We develop computational models that solve tasks directly within their mutable state, with dynamics governed by local cell interactions. These interactions are parametrized by a neural network serving as the cellular automaton's rules. The two key components of this neural network are its perception function and update function:

Perceive Function: The perceive function gathers information about each cell's neighborhood through learnable convolution filters applied to the immediate vicinity within the mutable state. This perception module transforms input state channels into a higher-dimensional perception vector, capturing relevant local spatial patterns. The kernel size, padding, and activation functions are configurable hyperparameters.

Update Function: The update function utilizes the perception vector and local hardware vector to update each cell's state. Our architecture employs an attention-based update module that calculates state update ΔS by conditioning perceived information P on an external input vector I, which encodes task-specific information or global context (in our implementation, representing local cell hardware).

1. Each cell receives a perception vector P (local spatial patterns) and a hardware vector I (its immutable state).
2. The hardware vector I activates different "computational modes" through an attention mechanism: $\alpha = \text{softmax}((I \cdot W_{\text{embed}})/T)$, where W_{embed} is a learned embedding matrix and T is a temperature parameter controlling the sharpness of the activation.

3. The perception vector P is simultaneously processed through N parallel pathways (implemented as MLPs), producing potential update vectors V_h for each pathway.
4. The final state update is computed as a weighted mixture of these pathways: $\Delta S = \sum_{h=1}^{N} \alpha_h V_h$, with the cell's state updated residually: $S_{t+1} = S_t + \Delta S$.

This design allows the NCA to adapt its behavior dynamically based on the local hardware—cells in input regions might activate different computational pathways than those in output regions or computational zones. The result is a flexible computational substrate where the same underlying rule can perform diverse operations depending on the hardware context.

3.4.3 *Hardware (Immutable State)*

As briefly explained in section 3.4.1, a core innovation in our approach lies in separating *mutable* and *immutable* parts of the computational state. This distinction separates the update model's role (which needs maximal generality and expressiveness) from task-specific hardware configurations that can be diverse and fine-tuned. We explore two different approaches for designing these specialized hardware configurations.

Monolithic hardware

Our first implementation optimizes task-specific parameters with the same spatial dimension as the computational state and a fixed number of hidden channels. This approach successfully trains the NCA on various tasks using specialized hardware for each. The optimized hardware configurations are visually interpretable, providing insights into the computation flow required for specific tasks (see fig. 3.3).

However, this approach lacks generalizability. For example, hardware optimized for matrix translation from bottom-left to top-right would require retraining for the opposite operation. It also compromises the inherently scale-free nature of NCA. An NCA trained for small-scale matrix multiplication would not be able generalize to larger matrices without hardware retuning. These limitations led us to develop a *modular hardware approach*.

Modular hardware

To address the limitations of monolithic hardware, we developed a modular and composable approach. Similar to how specialized components on a motherboard emulate desired behavior, we train three purpose-specific hardware *components*:

1. An input embedding vector specifying cells that receive inputs in the computational task. Cells whose mutable state receives inputs include this vector in their immutable state.

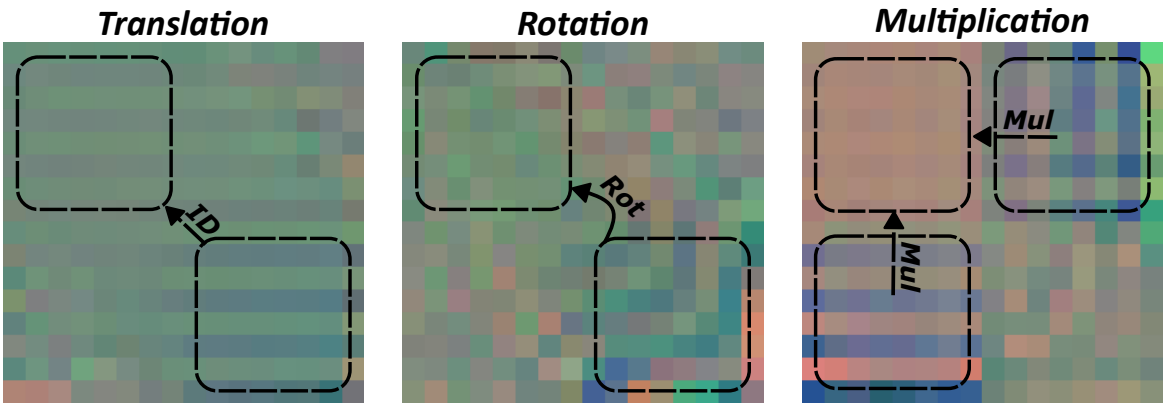


Figure 3.3: Monolithic hardware configurations for 3 different sub-tasks. We plot a PCA projection of the hidden channels to be able to display them as RGB. Colors thus does not have direct functional relevance. Overlayed are schematics of the (fixed) input-output transformations that each hardware was optimized on.

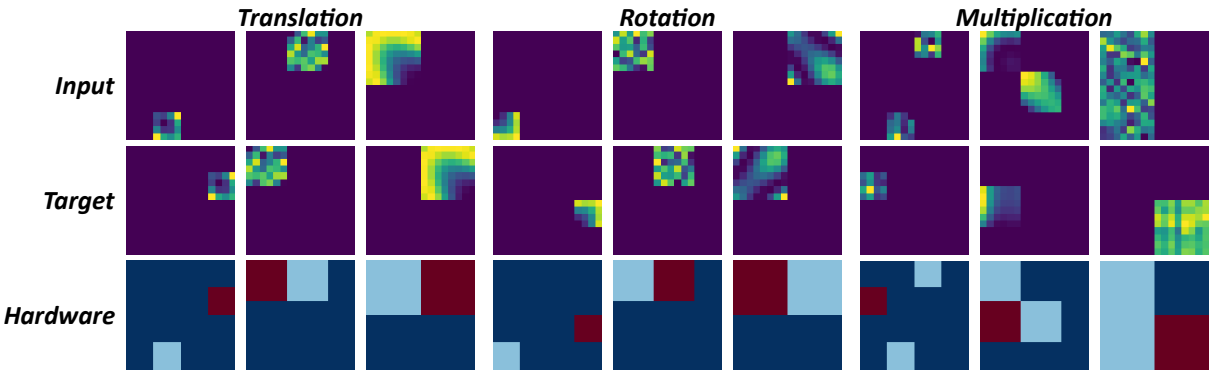


Figure 3.4: Modular hardware configurations for 3 different sub-tasks, and (now varying) different matrix sizes and placement.

2. An output embedding vector marking cells that will serve as output during a specific task.
3. A task embedding vector enabling the NCA to recognize the type of input-output transformation required. This learned vector is added to every cell's hardware state.

These three core components are then manually assembled for each task instance to create specific task examples. The resulting immutable state remains accessible to the update rule throughout an experiment. This modular approach balances the scale-free nature of NCA with the need for local heterogeneous substrate to perform diverse computational tasks. We also demonstrate that this enables zero-shot generalization, allowing the NCA to perform unseen task configurations and even composite task chaining (see section 3.5.3).

3.4.4 *Tasks*

To train robust and versatile Neural Cellular Automata (NCA) models capable of performing general computation, we implement a flexible framework of matrix-based operations. These tasks, exercise different computational capabilities and test the NCA's ability to process, transform, and route information across the grid.

MATRIX OPERATIONS The core of our framework is a set of fundamental matrix operations that represent different computational primitives:

- **Identity Mapping**: Reproducing an input matrix at a different target location. This tests information preservation and signal routing across the grid.
- **Matrix Multiplication**: Given input matrices A and B, compute $C = A \times B$. This tests the NCA's ability to perform non-local computations requiring information integration across regions.
- **Transposition /Rotation**: Given input matrix A, compute $B = A^T$ (or alternatively $B = A^T_{flipped}$ representing a 90 degree rotation in the plane) . This evaluates spatial information routing capabilities and geometric understanding.

DIVERSE INPUT DISTRIBUTIONS To prevent overfitting to specific input patterns, we employ matrices with varied statistical properties:

- **Uniform and Gaussian Distributions**: Randomly distributed matrix values test general processing capabilities.
- **Spatially Correlated Patterns**: Self-similar patterns with spatial correlations test the NCA's ability to process structured information and recognize spatial relationships.
- **Sparse Representations**: Matrices where most elements are close to zero test the NCA's efficiency in handling information sparsity.

FLEXIBLE PLACEMENT AND SIZE To ensure the NCA develops robust computational abilities that generalize beyond fixed spatial arrangements, we include in the task framework:

- **Dynamic Placement**: Input and output matrices can be positioned at different locations within the grid, preventing the NCA from memorizing fixed spatial patterns and forcing it to develop general computation mechanisms.
- **Variable Matrix Sizes**: Tasks involve matrices of different dimensions, from small to large relative to the grid size, testing the NCA's ability to scale its computations and adapt to varying information densities.

- **Multiple Inputs and Outputs**: Tasks can involve multiple input and output matrices distributed throughout the grid, requiring the NCA to coordinate information flow between different regions and perform parallel processing.

(This flexible task placement scheme is only usable in conjunction with a *modular* hardware configuration, as discussed in section 3.4.3)

Through this comprehensive framework, the NCA develops general computational capabilities that are robust to variations in task type, input distribution, matrix size, and spatial arrangement.

3.4.5 *Training*

To equip our Neural Cellular Automaton (NCA) with the capacity to perform a diverse range of computational tasks, we utilize a joint training framework. This approach simultaneously optimizes a single, shared NCA rule across a collection of distinct tasks instances. Each task is defined by a specific objective (an operation to be made on the inputs), and one task instance is typically represented by an initial grid state S_0, a desired target final state S_{target}, and often a mask M indicating the regions of the grid relevant for evaluation.

During training, batches containing instances from various tasks are sampled. For each instance, the NCA model evolves the initial state S_0 over a defined number of discrete time steps T_{steps} to produce a final state S_{final}. To enhance stability of the NCA, final states used to compute the loss are chosen at random between T_{steps} and $T_{steps} - T_{steps}//4$.

A loss function (commonly a masked error metric such as MSE), quantifies the discrepancy between the achieved final state and the target state within the relevant regions defined by the mask M.

Gradient-based optimization is employed to minimize this loss. Parameters associated with the shared NCA rule and any shared IO hardware components (in the *modular* hardware case) are updated based on gradients aggregated across all tasks within the batch, promoting the learning of general-purpose computational primitives. Parameters belonging to task-specific modules (*monolithic* hardware, or task components in the modular case) are updated using only the gradients derived from their corresponding task instances, enabling specialized behavior. This joint optimization process encourages the emergence of a versatile NCA capable of executing multiple computational functions through its learned local dynamics, dynamically adapting its behavior based on the presented task hardware.

3.5 EXPERIMENTS AND RESULTS

3.5.1 *Task Training*

Joint Training

In a multi-task training setup, our Neural Cellular Automata successfully master various matrix operations simultaneously through a shared update rule architecture combined with task-specific hardware components. Our findings demonstrate that a single NCA can develop general computational principles that apply across different matrix tasks while maintaining the specialized parameters needed for each specific operation.

The multi-task learning capability reveals the fundamental computational versatility of NCA. By simultaneously learning to perform diverse operations such as matrix multiplication, translation, transposition and rotation within a unified framework, the model demonstrates mastery of a complete algebra of matrix operations—the essential building blocks for more complex computation.

This multi-task foundation directly enables more sophisticated composite applications, such as our MNIST classifier emulation (section 3.5.2). The ability to decompose complex operations into smaller matrix tasks and process them through the same underlying cellular mechanism demonstrates a pathway toward increasingly complex computation. By establishing that NCA can reliably perform these fundamental operations, we provide the essential building blocks for future work on more elaborate composite tasks, including full neural network emulation, algorithmic reasoning, and potentially even more advanced computational models implemented entirely within the cellular substrate.

Downstream Tasks Fine-tuning

A key advantage of our architecture emerges once the NCA is pre-trained: new tasks can be accommodated by fine-tuning only the hardware configurations while keeping the core CA update rules frozen. This approach significantly reduces computational requirements for adaptation to novel tasks. In our experiments, fine-tuning hardware alone increases training speed by a factor 2, compared to full model retraining. A more comprehensive comparison of joint training vs fine-tuning is nevertheless needed.

3.5.2 *MNIST Classifier emulation*

We demonstrate a practical downstream application by using our Neural Cellular Automata (NCA) to emulate an entire neural network directly in its computational workspace. Specifically,

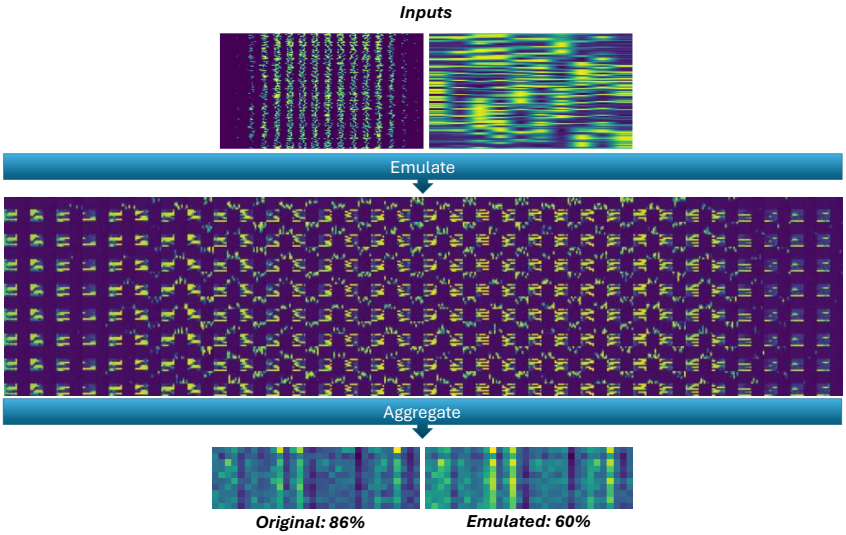


Figure 3.5: NCA emulates a neural network. Inputs show a batch of flattened MNIST images, alongside the weight matrix of a pre-trained single-layer linear classifier. We decompose this matrix multiplication into sub-blocks, that can be directly emulated in parallel by the NCA. Results are fetched from NCA states and aggregated back into logits (figure shows the first 32 outputs only for readability). We compute accuracy by taking the logits argmax per batch and comparing with labels.

we emulate a single-layer Multi-Layer Perceptron (MLP) solving the MNIST digit classification task.

First, we pre-train a simple linear feedforward network to classify MNIST digits with sufficient accuracy. This classifier uses a single weight matrix without bias terms, where inference requires only a matrix multiplication between the flattened input images and the weight matrix, followed by an argmax operation to determine the predicted digit. Our NCA model was pre-trained as well, on smaller 8×8 matrix multiplication tasks. While we could hope for generalization to larger matrices, operations of the scale required for MNIST classification (784×10) would exceed the capacity of what can be performed by such a model. To address this limitation, we implement block-matrix decomposition, fragmenting the classification of MNIST images into multiple smaller 8×8 matrix operations that fit within the NCA's state constraints. The resulting decomposed operations can be executed directly within the NCA's computational state without requiring task-specific fine-tuning, demonstrating the robustness of our approach to novel matrix distributions. The NCA processes each block multiplication operation in parallel, after which we aggregate the results to reconstruct the complete classification logits. When evaluating performance, we compare the predictions and accuracy of our NCA-based emulation against the original classifier. While we observe some accuracy degradation due to error propagation across numerous sub-operations, the model still achieves respectable performance (around 60% accuracy for the emulated classification compared to 84% for the original, with predictions agreeing around 69% of the time). We argue that this is providing empirical evidence that neural network emulation via NCA is feasible.

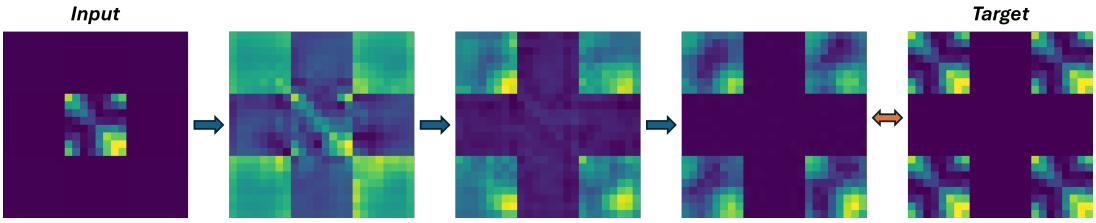


Figure 3.6: Examples of an Out of Distribution Task: the NCA needs to distribute a matrix in all corners, in a larger grid than the one seen during training.

This research has significant implications for analog and physical computing. If our NCA's update rules were implemented as physical state transitions, this would represent a pathway toward physical neural network emulation without reverting to binary-level operations. The ability to operate directly at the level of matrix operations using hardware specifically designed for this computational paradigm could offer substantial efficiency and performance improvements over conventional digital approaches.

3.5.3 *Future directions: task composition and neural compiler*

The modular hardware configuration we developed enables straightforward creation of out-of-distribution tasks through component composition. This flexibility allows us to design novel computational scenarios that the NCA was not explicitly trained on, yet can still execute successfully. For instance, we can implement data distribution patterns by duplicating a central matrix into multiple corners using the matrix translation task embedding and multiple target tiles fig. 3.6. This simple example demonstrates how our architecture supports operations beyond the training distribution without requiring additional training. This framework opens the path toward complex composite tasks created through sequential chaining of primitive operations. Consider the following multi-step procedure (fig. 3.7 bottom panel):

1. Start with a input matrix and distribute copies to two corner positions using target tiles.
2. Replace the current hardware configuration with new parameters that redefine these targets as inputs, then perform matrix multiplication towards a third corner.
3. Update the hardware again to rotate the resulting matrix and return it to the original position.

While such composite sequences may not appear useful on their own, they demonstrate a critical capability: the NCA can execute complex algorithmic workflows through sequential reconfiguration of its hardware parameters. This capability lays the groundwork for more sophisticated computational reasoning and abstraction.

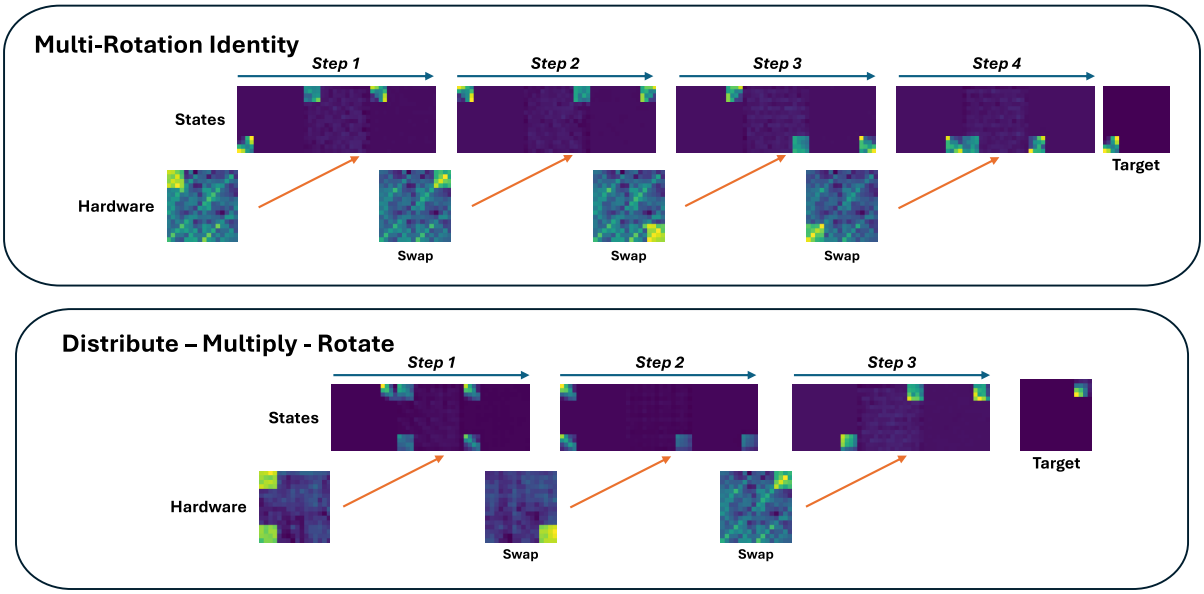


Figure 3.7: Illustrating composite computational tasks using modular hardware configurations. Top panel: Multi-Rotation Sequential rotations resolving in the identity function. Bottom panel: Distribute-Multiply-Rotate task, showing a three-step process where matrices are distributed, multiplied, and rotated to achieve the target state. Each step shows both the hardware configuration and corresponding computational states.

This composite task chaining also highlights the crucial role of stability in achieving composable computations. When outputs produced within the NCA's computational state serve as inputs for subsequent operations, establishing stable representations becomes essential. In parallel to biological systems, where homeostasis maintains internal equilibrium despite external changes, NCA require a form of computational homeostasis to maintain reliable state representations between operations. Similarly, analog computers may require homeostasis to successfully implement extended chains of tasks and computations without degradation of information.

We propose that this sequential operation model suggests a compelling dual-timestep approach to neural compilation: At the neuronal timestep, the NCA's mutable state evolves according to its update rules, creating the fundamental dynamics of computation. At the compiler timestep, hardware parameters are reconfigured to provide task abstractions and high-level procedural steps. This separation of concerns, where fast neuronal dynamics handle computation while slower hardware changes control program flow, mirrors classical computer architecture but within a continuous, differentiable substrate. As this approach matures, it could enable direct compilation of algorithms into neural cellular automata, combining the flexibility of neural networks with programmatic execution. This would possibly be facilitated by a better task-abstraction and hardware generation that we detail in the next section.

Graph-based hardware hypernetwork

Finally, building upon the limitations of the previous hardware approaches, we are curently developing a more principled graph-based hardware generation framework that offers significant improvements in both flexibility and scale-invariance. This (WIP) approach leverages a task representation abstraction where computational operations are modelled as a graph, with nodes representing input and output regions and edges encoding specific transformations between them. At the core of this framework is a Hardware Meta-Network consisting of two main components: a Graph Neural Network (GNN) encoder and a coordinate-based hypernetwork.

The GNN processes a task graph structure, where nodes contain normalized spatial information about input and output regions, and edges represent specific operations (e.g., matrix multiplication, rotation) to be performed between regions. Through multiple message-passing layers, the GNN distils this graph representation into a fixed-dimensional latent task vector that captures the essential computational requirements of a single task instance problem.

This latent representation then conditions a coordinate MLP hypernetwork that generates hardware vectors for every spatial location in a scale-free manner. The hypernetwork leverages positional encodings to create spatially varying hardware patterns that guide the NCA's computational dynamics across the grid. Crucially, this approach maintains exact spatial invariance: task specifications are normalized relative to grid dimensions, which should enable the generated hardware to automatically adapt to different grid sizes and region placements without retraining.

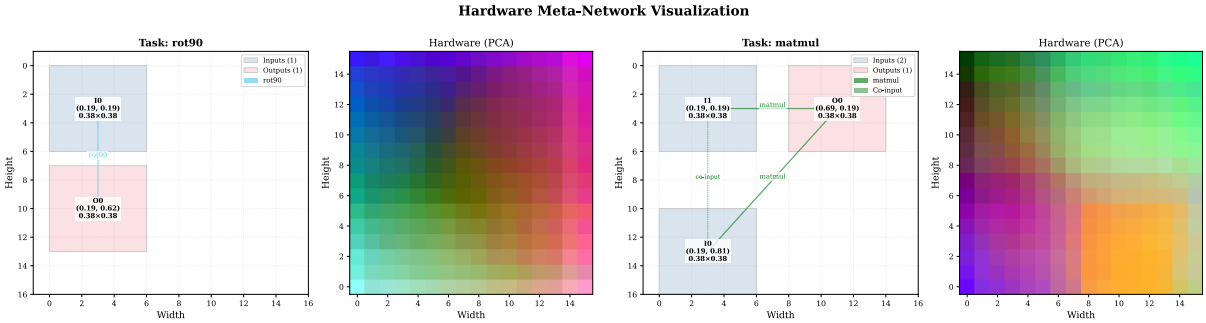


Figure 3.8: Graph-based tasks representations and GNN-based hypernetwork for hardware generation

This graph-based representation provides an intuitive interface between human-specified computational tasks and the continuous NCA substrate. Users can define tasks through a natural graph specification (inputs, outputs, and operations), and the meta-network translates these specifications into appropriate hardware configurations. This approach would effetively improve on the concept of a compiler between human intent and the NCA's computational capabilities, allowing for better definition of task chaining and temporal hardware evolution. Moreover, the graph structure can enable rich extensions beyond our initial implementation. Tasks can be ordered through edge attributes, allowing sequential execution planning. Dynamic

hardware reconfiguration becomes possible by modifying the task graph over time, creating a secondary dynamics layer that complements the fast neural dynamics of the cellular automaton itself. This hierarchical temporal structure, where fast neural dynamics implement local computations while slower hardware dynamics guide algorithmic flow, mirrors the dichotomy in traditional computing architectures between clock-cycle operations and higher-level program execution. Doing so within a unified differentiable framework may ultimately enable more efficient, adaptable continuous computational paradigms.

3.6 CONCLUSION

The exploration of universal computation within cellular automata has historically been confined to discrete systems, where models like Conway's Game of Life and Elementary Cellular Automata have demonstrated the remarkable ability to emulate Universal Turing Machines. However, extending this capability to continuous cellular automata presents significant challenges, primarily due to the absence of discrete states and the inherent instability of smooth, analog dynamics. In this work, we have taken pragmatic first steps toward overcoming these hurdles by leveraging NCA as a substrate for developing universal computation in a continuous domain. By employing gradient descent to train NCA rules, we have demonstrated a pathway to sculpt complex computational behaviours without the need for manual rule design, shifting the burden of discovery from human ingenuity to machine learning.

Our results illustrate that NCA can successfully encode fundamental computational primitives, such as matrix multiplication and inversion, and even emulate a neural network capable of solving the MNIST digit classification task directly within its state. These findings suggest that NCA can serve as a bridge between traditional computing architectures and self-organizing systems, offering a novel computational paradigm that aligns closely with analog systems. This linkage is particularly promising for designing efficient computational frameworks for AI models operating, where energy efficiency and robustness are paramount. Rather than training entirely new rules for each task, our approach hints at the possibility of discovering optimal hardware configurations that exploit the fixed physical laws governing these substrates, enabling meaningful computations with minimal overhead.

Looking forward, we believe this work lays the groundwork for transformative advancements in computational science. By automating the discovery of general-purpose computers within diverse physical implementations, NCA could revolutionize how we harness novel materials and systems for computation, potentially leading to ultra-efficient analog hardware systems or computational paradigms that scale linearly with resource demands. While challenges remain — such as stabilizing continuous dynamics for reliable symbolic encoding and scaling these systems to more complex tasks—the potential of NCA to unlock universal computation in continuous cellular automata opens new avenues for exploration. Ultimately, this research not

only advances our understanding of computation in continuous dynamics but also paves the way for the next generation of adaptive, energy-efficient computing technologies.

4

SELF-ORGANISING DIGITAL CIRCUITS

Narrative Preamble

If Chapter 3 explored the computational potential of local rules, this chapter explores their adaptive potential. We move closer to the biological reality of a system that must not only compute, but also maintain itself and adapt to changes. Here, we utilize the principles of self-organization as a "meta-learning" mechanism. Instead of the NCA being the computer, it becomes the optimizer. We investigate a system where local rules navigate and configure a functional digital circuit, effectively replacing global backpropagation with a decentralized, graph-based optimization process. This architecture has implications for true scalable intelligence: a system where the "learning rule" is itself a local, robust policy. By demonstrating how such a system can recover from physical damage and reconfigure itself without global supervision, we provide a concrete instantiation of the "adaptive resilience" that characterizes biological intelligence, pointing the way toward systems that truly know how to learn and adapt.

Publication Context

This chapter presents ongoing, unpublished collaborative work with Marcello Barylli from the group of Prof. Sebastian Risi at the IT University of Copenhagen. The project originated from a concept by Alexander Mordvintsev, at the 2025 Capocaccia workshop.

Contributions

Alexander Mordvintsev conceived the initial project idea and developed the initial software: differentiable Boolean circuits and the GUI for observing backpropagation behaviour. The author and Marcello Barylli formalized the architectural shift to use a Transformer as a meta-optimizer. The author led the development of the NCA framework, implementing graph-based models, the meta-learning and pool infrastructure, and conducted the bulk of the "growth" wiring experiments [1]. Marcello conducted the primary experiments regarding functional maintenance. The author also acknowledges Maxence Faldor whose vision on residual dynamic systems, NCA as a general graph learning-rule and the perceiver architecture greatly influenced the later stages of this research

1 Data/Code + **Circuit Visualization** available here

4.1 ABSTRACT

Fault tolerance in classical computing has traditionally relied on static strategies like hardware redundancy and error-correcting codes. Biological systems, in contrast, exhibit *adaptive plasticity*, maintaining function through dynamic re-organisation around damage. Inspired by this principle, we introduce a framework for self-organising digital circuits that replaces rigid redundancy with learned, autonomous self-repair. We extend the Neural Cellular Automata (NCA) paradigm from pattern formation on grids to functional logic generation on arbitrary graphs. Our architecture employs a topology-masked Transformer as a decentralized meta-optimizer that learns to configure the Look-up Tables (LUTs) of Logical Gate Networks (LGNs). Unlike standard NCA which regenerate a fixed target state, our model learns a functional local policy that navigates the Boolean search space to satisfy a computational task. We demonstrate that this policy can self-assemble functional circuits from scratch and, crucially, rapidly *re-route* logic around permanent, previously unseen hardware faults. This work bridges the principles of biological self-organisation with the practical domain of digital hardware, offering a blueprint for resilient systems capable of graceful degradation and real-time adaptation.

4.2 INTRODUCTION

Motivation: From Prescriptive Redundancy to Adaptive Plasticity

Robustness is a central tenet of modern informatics. Decades of research have yielded systems capable of impressive fault tolerance, utilizing mechanisms such as Error Correcting Codes (ECC), modular redundancy, and sophisticated fallback protocols to ensure graceful degradation (von Neumann, 1956; Woods and Lightbody, 2008). However, these engineered successes typically rely on *anticipation*: resources are pre-allocated and failure modes are modelled in advance. When a system encounters damage that exceeds its redundancy budget or defies its failure model, it might fail brittely. Biological intelligence offers a complementary paradigm: *adaptive plasticity*. Natural automata, such as the mammalian cortex, do not rely solely on static backups. Instead, they exhibit the ability to dynamically re-purpose surviving components to compensate for injury, a phenomenon known as cortical remapping (Nudo, 2013). This form of resilience is not pre-programmed, but emergent: the system "learns" its way around the damage. Inspired by these biological principles, and by early von Neumann's assumption that systems should "operate across errors" (von Neumann, 1966), we seek to endow digital hardware with similar capabilities. This work introduces a framework for self-organising digital circuits that complements traditional redundancy with dynamic, learned self-repair. In this sense, it is a new take on the idea of self-repairing digital hardware, which has already been considered as

a robustness mechanism, especially in FPGAs (Carter et al., 1986; Gokhale et al., 2004; Woods and Lightbody, 2008)

The Evolution of Automata: From Fixed Rules to Learned Dynamics

The study of Cellular Automata (CA) has long served as a fertile ground for investigating the essence of computation and complexity. Fundamentally, a Cellular Automaton is a discrete computational system where a grid of cells, each holding a specific state, evolves in synchronized steps. The evolution of each cell follows the same local rule that dictates how every cell updates based on its immediate neighbours on the grid. From simple local rules, CAs can generate remarkably intricate patterns and have been proven to be Turing complete (Cook, 2004). However, CAs suffer from a significant control problem: while it is easy to observe complexity arising from pre-determined rules (Wolfram, 2002), explicitly engineering specific local rules to achieve a desired global outcome is notoriously difficult. Attempts to extend these systems into the continuous domain, such as Lenia or SmoothLife (Chan, 2019; Rafler, 2011), introduced life-like fluidity but further complicated the stability and control of the system's outputs.

Recently, Neural Cellular Automata (NCA) have emerged as a paradigm shift that bridges the gap between self-organising systems and differentiable learning. In an NCA, the update rule is no longer hard-coded; it is a parameterised neural network (Mordvintsev et al., 2020). Each cell in the system utilizes a "perception" network to gather information from its local neighbourhood (typically on a grid), and an "update" network to compute an incremental change to its state. Because these operations are differentiable, the dynamics of the system can be "steered" via gradient descent towards specific goals. Traditionally, NCA have been utilized as models of morphogenesis: for example growing target images by treating the final cell states as RGB values. In these architectures, cells also maintain a vector of "hidden states," allowing the local rule to encode arbitrary information required to coordinate global structure. This distinction already let us differentiate between part of the cells' states that are "functional" (such as the RGB channels) and other that are "latents".

Beyond the Grid: Growing Function over Form

While foundational, standard NCA remain topologically constrained to rigid grids and functionally constrained to pattern formation. To overcome these limitations, recent research has sought to extend the NCA paradigm into more flexible and general models of computation. First, the topological constraint: by reinterpreting the NCA update step as a message-passing operation on a graph, it is possible to generalise the concept of "neighbourhood" from spatial proximity

to topological connectivity (Grattarola et al., 2021). In this context, Graph Neural Networks (GNNs) provide the natural machinery for self-organisation: nodes "perceive" their neighbours via messages sent along edges, aggregate this information, and update their internal state (in an incremental manner, to stay true to the core NCA dynamics). Second, the functional constraint: traditionally, NCA states represent the target pattern itself (e.g., RGB values), but this can also be extended. We've seen in Chapter 3 that we can move closer to general computation by interfacing continuous NCA with matrix tasks. While this demonstrated that NCA could learn a dynamic *process* (e.g., matrix multiplication) rather than a static final state, it still conflated the CA's state with the computational data: effectively, the cells states were still holding the final result of the computation (the matrix values). We propose here a more fundamental shift: viewing the cellular state not as the data to be processed, but as the *functional substrate* that performs the subsequent processing.

To implement this, we utilise digital Boolean circuits as our substrate. Adopting the framework of Differentiable Logical Gate Network (LGN) (Petersen et al., 2022), we instantiate graphs where we treat nodes as gates defined by Lookup Tables (LUTs) and edges as functional wires. Here, the "state" of a node is not a pixel colour or a matrix value, but the functional logic (the LUT configuration) it implements. By recursively applying a decentralised message-passing scheme, our NCA then acts as a meta-optimiser: it steers the substrate not toward a visual shape, but toward a configuration that is later-on able to implement a specific logic task (e.g., binary multiplication, addition, XOR etc).

The topology-masked Transformer

As we've just discussed, GNNs are well suited to update graphs residually, in a manner that extends the NCA paradigm to act on unconstrained topologies. Our architecture, however, employs a Transformer as the update rule. To see why this is natural, it is helpful to view self-attention through the lens of message passing. In a standard self-attention layer, each token updates its representation by computing a weighted sum over all other tokens, where the weights are determined by learned key-query interactions. This is, in essence, a message-passing operation on a *complete graph*: every token sends a message to every other, and the receiver decides—via the attention weights—how much to listen to each sender. The crucial difference with classical GNNs is that the aggregation function is not fixed (e.g., sum or mean); instead, it is *learned and input-dependent*, allowing the model to dynamically route information based on content.

Nonetheless, the graph-NCA paradigm requires strict locality: a node should only exchange information with its direct graph neighbours at each step. We enforce this by applying a *topology mask* M to the attention matrix, where $M_{ij} = 1$ if and only if a wire connects gates i and j (or $i = j$), and $M_{ij} = 0$ otherwise. Masked-out entries receive $-\infty$ before the softmax, ensuring

zero attention weight. This single modification transforms the complete-graph communication pattern into message passing along the circuit's edges: each gate attends only to its wired neighbours and itself, with the key-query mechanism learning *how* to weight and combine these neighbour messages at each step. We note that this is slightly distinct from "Graph Transformer" architectures (Dwivedi and Bresson, 2021), which typically introduce graph-specific modifications such as structural encodings or edge features within the attention computation itself. Our approach is deliberately minimal: the topology enters *only* through the attention mask, and all expressivity comes from the standard Transformer machinery.

Furthermore, this design aligns with recent advances in meta-learning optimisers and general-purpose in-context learning (Kirsch et al., 2024; Kirsch and Schmidhuber, 2022), which have demonstrated that Transformers can be meta-trained to internalise learning algorithms—effectively replacing explicit optimisers (like SGD) with a forward-pass policy. Similarly in our system, the topology-masked Transformer effectively becomes the optimiser: a single, shared-weight attention block is applied recurrently for many steps, gaining representational depth through *time* rather than through stacked layers. It learns a robust, local policy that navigates the landscape of Boolean configurations to find functional solutions.

Contributions and Key Findings

By applying these self-organising principles to digital hardware, we present a concrete instantiation of "adaptive resilience". Unlike standard backpropagation, which requires global knowledge of the circuit and differentiable components to repair damage (Sunada et al., 2025), our system relies solely on local forward passes of the NCA. This decoupling could allow for potential deployment on re-programmable hardware, such as FPGAs, where the "repairing agent" can operate efficiently on-chip. We demonstrate that this meta-learned network can not only discover functions from scratch but also rapidly reconfigure circuits to recover from permanent, unseen physical damage, pointing the way toward autonomous systems that truly know how to learn and adapt.

4.3 METHODS

4.3.1 *Differentiable Logic Gate Networks*

To enable gradient-based optimisation of digital logic, we adopt the LGN framework (Petersen et al., 2022). A LGN is a Directed Acyclic Graph (DAG) where nodes represent logic gates and edges represent wires carrying bit signals. Unlike standard digital circuits where gates have

fixed truth tables (e.g., AND, XOR), our gates are parameterised by learnable Lookup Tables (LUTs).

Table 4.1: Standard Look-Up Tables (Truth Tables) for 2-Input Logic Gates

Inputs		**Gate Output**					
A	B	**AND**	**OR**	**XOR**	**NAND**	**NOR**	**L**
0	0	0	0	0	1	1	l_{00}
0	1	0	1	1	1	0	l_{01}
1	0	0	1	1	1	0	l_{10}
1	1	1	1	0	0	0	l_{11}

Continuous Relaxation of a gate: Standard Boolean operations are non-differentiable. We relax this by representing binary signals as continuous probabilities $x \in [0, 1]$. For a gate with arity k (number of inputs), the LUT is a tensor of 2^k learnable logits (such as the L column of table 4.1). The "soft" output of a gate is computed via multilinear interpolation, effectively traversing the LUT as a probabilistic Binary Decision Diagram (BDD). For a gate with inputs $\mathbf{x} = [x_1, \ldots, x_k]$ and LUT parameters $\mathbf{L}$, the forward pass recursively interpolates the table. For instance, with a 2-input gate, the first input x_1 interpolates between the two halves of the LUT to produce an intermediate tensor L':

$$L' = (1 - x_1) \cdot [l_{00}, l_{01}] + x_1 \cdot [l_{10}, l_{11}]$$

The second input x_2 does the same in L' and computes the final scalar output:

$$y = (1 - x_2) \cdot L'[0] + x_2 \cdot L'[1]$$

This operation is fully differentiable, allowing gradients from a loss on output y to flow from the output back to the individual LUT entries. In practice, parameters are logits ℓ, from which we get look-up table entries L before functional execution. During training, we get $L = \sigma(\ell)$ with σ being the sigmoid function. During inference however, the circuit can operate in "hard" mode by rounding LUT entries to $\{0, 1\}$ with $L = \Theta(\sigma(\ell))$, with Θ being a hard rounding function, thus recovering discrete Boolean logic.

Global Circuit Architecture and Execution: The boolean gates are organised into a deep, feed-forward network of sequential layers. A critical architectural distinction exists between the global circuit inputs and the local inputs required by each gate. The circuit receives a global input vector $\mathbf{x} \in [0, 1]^{N_{in}}$, while individual gates operate with a fixed arity k (fan-in). Signals propagate through the network via "wires," which, although they represent physical connections, are implemented as integer indices selecting specific bits from the previous layer. For example, consider a first layer of two arity-2 gates (G_1, G_2) processing a 4-bit global input

$\mathbf{x} = [x_0, \ldots, x_3]$. The wiring mechanism might assign indices $(0, 1)$ to G_1 and $(2, 3)$ to G_2. Consequently, G_1 computes $LUT_1(x_0, x_1)$ while G_2 independently computes $LUT_2(x_2, x_3)$. This indexing allows for arbitrary connectivity and signal duplication. The collective outputs of these gates then form a vector, which serves as the input for the subsequent layer.

Circuit Dimensioning: The number of gates in each layer is systematically computed to ensure both sufficient computational capacity and structural compatibility between adjacent layers. For a circuit with N_{in} input bits, N_{out} output bits, and a gate arity of k, the standard hidden layers are expanded by a width factor W (set to 2 in our experiments), resulting in layers of size $N_{in} \times k \times W$. Crucially, to satisfy the exact input requirements of the final layer, the last hidden layer is strictly dimensioned to $N_{out} \times k$ gates. For our 12-bit tasks with arity $k = 4$, this calculation naturally generates the three-hidden-layer architecture of sizes $(96, 96, 48)$ used throughout our experiments.

Boolean Tasks: We evaluate the meta-learner on three distinct tasks operating on 12-bit global inputs. For any boolean task, this 12-bit input space thus yields a dataset of exactly 4,096 input–output pairs. While this constitutes a relatively small total domain, we strictly reserve a split of 256 examples as a held-out test set to ensure the policy learns the underlying generative function rather than merely memorising the training data. The final circuit output layer always consists of 12 bits:

- **Split Multiplication:** The 12-bit input is split into two 6-bit integers. The network computes their product, which perfectly occupies the 12-bit output capacity without overflow.
- **Split Addition:** The input is similarly split into two 6-bit integers to be added. Because the sum requires only 7 bits (accounting for the carry), the remaining 5 highest-order output bits naturally act as zero-padding.
- **Bit Reversal:** The network must structurally route signals to reverse the input array, mapping the i-th input bit to the $(11 - i)$-th output bit.

Initialization: To ensure early gradient flow, circuits are initially configured as "no-ops" (soft wires) rather than with random logic. For each gate of arity k, we initialise the look-up table logits such that the gate acts as an identity function for one of its inputs, selected by cycling through the inputs across the layer. These base logits are scaled to ± 3.0 (meaning the resulting luts entries are $\approx \pm 1$) to create strong initial pass-through behaviour, while a small amount of additive Gaussian noise is injected to break symmetry and facilitate divergence during optimisation.

4.3.2 *Graph Representation of Circuits*

To apply self-organising principles, we lift the circuit instance into a graph representation suitable for processing by a Graph Neural Network. Given a circuit with a set of gates G and wires W, we construct a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where each node $v_i \in \mathcal{V}$ corresponds to a gate or an input node.

Node State: The state of each node v_i is a vector concatenation of functional and auxiliary features:

$$\mathbf{s}_i = [\boldsymbol{\ell}_i, \mathbf{m}_i, \mathbf{p}_i]$$

where:

- $\boldsymbol{\ell}_i \in \mathbb{R}^{2^k}$ is the functional LUT logit vector defining the gate's logic.
- $\mathbf{m}_i \in \mathbb{R}^{d_{hidden}}$ is a latent memory vector allowing the meta-learner to store recurrent state across timesteps (hidden states). Unless specified, $d_{hidden} = 64$
- $\mathbf{p}_i$ is a sinusoidal positional encoding of the gate's normalised depth fraction $\frac{d_i}{D}$, where d_i is the layer index and D the total number of layers. By encoding depth as a ratio rather than an absolute index, this representation remains invariant to circuit scale: a gate at the midpoint of a 4-layer circuit receives the same encoding as one at the midpoint of a 10-layer circuit.

Optionally, a scalar local feedback signal $\mathbf{r}_i$ (described in hte next section) can be appended to provide the meta-learner with per-node error context, yielding $\mathbf{s}_i = [\boldsymbol{\ell}_i, \mathbf{m}_i, \mathbf{p}_i, \mathbf{r}_i]$.

Connectivity: The graph topology is strictly dictated by the circuit wiring. While fixed-topology experiments utilise a static diagram, our random topology regime generates connectivity dynamically. For a layer of gates, we construct connections by randomly permuting the indices of the previous layer's outputs. This guarantees a uniform fan-out distribution while preserving the feed-forward DAG structure. To allow gradient and error information to flow upstream during meta-learning, we enforce bidirectional message passing: every forward logic wire $A \to B$ implies a backward message-passing edge $B \to A$. This ensures local update rules can propagate functional context backward through the topology.

4.3.3 *Meta-Learning Architecture: Topology-Masked Transformer*

We parameterise the update rule using a single-block Transformer that operates on the entire circuit simultaneously. Unlike standard GNNs that iterate over edges, we flatten the circuit graph into a set of N tokens, allowing us to leverage standard self-attention machinery while strictly enforcing the circuit's connectivity constraints. In the following paragraphs, any matrix $\mathbf{M}$ normalized via layer-normalization (Ba et al., 2016) is denoted as $\hat{\mathbf{M}}$ for conciseness.

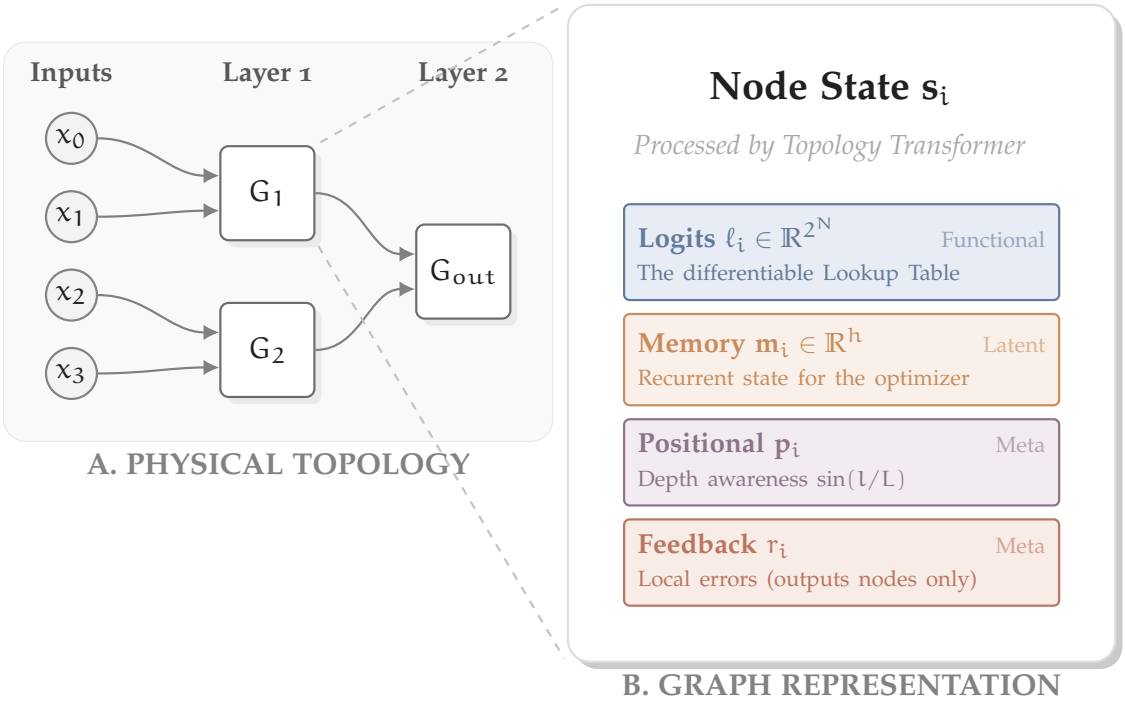


Figure 4.1: **Circuit Architecture and Graph Representation. (A)** The physical topology of the Differentiable Logic Gate Network. Gates (nodes) are arranged in layers and connected by wires (indices). **(B)** The graph representation of a single node state $\mathbf{s}_i$. This state vector concatenates the functional parameters (Logits) with the meta-learning variables (Memory, Position, Feedback) that the Graph Transformer utilizes to optimize the circuit.

Feature Projection and Tokenization. The node states $\mathbf{S} \in \mathbb{R}^{N \times d_{in}}$ (concatenating logits $\boldsymbol{\ell}_i$, memory $\mathbf{m}_i$, position $\mathbf{p}_i$, and feedback r_i) are normalised and projected into a latent space (size $d_{attn} = 128$) using feature weights W_{in}:

$$\mathbf{Z}^{(0)} = \hat{\mathbf{S}} W_{in}^{T} \in \mathbb{R}^{N \times d_{attn}} \tag{4.1}$$

Masked Attention Block. The latents are refined by a single Transformer block whose attention is restricted to wired neighbours via a binary mask $M \in \{0, 1\}^{N \times N}$, the first part of which is the **MSA** (masked self-attention), followed by a two layer **MLP** with GeLu activation. We adopt **Pre-LN** normalisation (Xiong et al., 2020) and **QK-normalisation** (Dehghani et al., 2023) to stabilise attention logits across recurrent steps, and **ReZero** gating (Bachlechner et al., 2020) (α scalars initialised at zero) so that the block initially acts as an identity

$$\mathbf{Z}' = \mathbf{Z}^{(0)} + \alpha_{\text{attn}} \cdot \text{MSA}\left(\hat{\mathbf{Z}}^{(0)}, \hat{\mathbf{Z}}^{(0)}, M\right) \tag{4.2}$$

$$\mathbf{Z}_{out} = \mathbf{Z}' + \alpha_{\text{ffn}} \cdot \text{MLP}\left(\hat{\mathbf{Z}}'\right) \tag{4.3}$$

Separate LayerNorms for queries and keys/values decouple their statistics. Crucially, the MLP is applied independently to each node's latent representation, ensuring that all cross-node communication is confined strictly to the masked attention step.

Output Updates. The output latents are decoded into parameter updates via linear heads, again ReZero-gated:

$$\Delta\boldsymbol{\ell}_i = \alpha_\ell \cdot \hat{\mathbf{Z}}_{\mathrm{out},i} W_\ell^T, \qquad \Delta\mathbf{m}_i = \alpha_m \cdot \hat{\mathbf{Z}}_{\mathrm{out},i} W_m^T \tag{4.4}$$

These are applied as residual updates: $\boldsymbol{\ell}_i \leftarrow \boldsymbol{\ell}_i + \Delta\boldsymbol{\ell}_i$ and $\mathbf{m}_i \leftarrow \mathbf{m}_i + \Delta\mathbf{m}_i$. Note that positional encodings $\mathbf{p}_i$ remain static throughout the recurrent pass, while the error feedback r_i is dynamically recomputed by the environment.

Per-Node Error Feedback. As described so far, the architecture has no direct information about how well the circuit performs on its task. For fixed-wiring experiments this is sufficient; however, when generalising across *random wirings*, we found it critically insufficient. We augment each output gate's state with a scalar feedback signal r_i—the average absolute residual across a batch of task evaluations:

$$r_i = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}} |y_i - \hat{y}_i(\mathbf{x})| \tag{4.5}$$

where $\hat{y}_i$ is the soft output of gate i on input $\mathbf{x}$. This signal is recomputed at every recurrent step; non-output gates receive $r_i = 0$. Empirically we find that this simple mechanism is essential for wiring-agnostic optimisation: the signal propagates upstream through recurrent attention steps, allowing interior gates to adjust in response to downstream error—a decentralised form of credit assignment. We note that this still leaves the meta-learner *blind* to the actual input–output data ; cross-attention to input and output tokens (in the spirit of Perceiver IO Jaegle et al., 2022) is a promising avenue for removing this limitation, and is the subject of ongoing work.

Recurrence as Depth. The architecture employs a *single* attention block applied recurrently for T steps:

$$\mathbf{s}_i^{(t+1)} = \mathbf{s}_i^{(t)} + \Delta\mathbf{s}_i^{(t)}, \quad t = 1, \ldots, T \tag{4.6}$$

where $\Delta\mathbf{s}_i^{(t)}$ is computed by the same shared-weight block at every step. This can be viewed as a T-layer weight-tied residual network. Expressivity, in this case, comes through *iterated refinement*: each step nudges each gate based on its current neighbourhood, and the accumulation of many such nudges achieves globally coordinated solutions. The topology mask in this sense imposes a strict "speed of light": information propagates at most one hop per step, so a gate k edges away requires $\geqslant k$ steps to exert influence. Global coordination thus emerges from iterated local interactions, a deliberate inductive bias faithful to the NCA paradigm, ensuring the policy is fundamentally *local* and that global solutions arise through repeated application of local rules over time.

Scale-Free Architecture. Because the Transformer operates at the node level with shared weights, its parameters are independent of circuit size—the same update rule applies to 20 or 200 gates, 3 or 10 layers. Only the topology mask M must be recomputed from the wiring. Combined with normalised positional encodings, this endows the architecture with *scale-freedom*: a trained policy can, in principle, be deployed on circuits of different size without retraining.

4.3.4 *Training Framework: Pool-Based Meta-Learning*

We treat the circuit optimisation problem as a dynamic system trained via Backpropagation Through Time (BPTT).

Pool Mechanism: As in the original NCA paper, we maintain a persistent graph **Pool** $\mathcal{P} = \{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$ containing K circuit instances that are continually optimized. By persisting state across many training steps, this mechanism favours policies that achieve and maintain homeostasis, as circuits must remain stable over potentially long durations. To ensure diversity and prevent stagnation, a portion of the pool is periodically reset, injecting "fresh" unoptimized circuits back into the process. The reset rate is calibrated such that circuits are optimized for approximately 128 steps on average before replacement. While this expected lifespan may appear short, we find that the stochastic nature of the pool creates a vital dual curriculum: the constant injection of fresh circuits forces the policy to learn rapid self-assembly ("growth"), while the "survivor" circuits that linger in the pool for much longer than the average require the policy to learn long-term stability ("homeostasis").

Inner Loop (Inference — "Growth"): At each training step, a batch of circuits is sampled from the pool alongside a batch of Boolean input–output pairs $(\mathbf{X}_{\mathbf{train}}, \mathbf{Y}_{\mathbf{train}})$ for the current task. The Transformer is then applied iteratively for T steps via a differentiable `scan` operation. Crucially, the circuit is functionally executed *at every step*: at each tick t, the updated LUT logits are extracted, the circuit is run on $(\mathbf{X}_{\mathbf{train}}, \mathbf{Y}_{\mathbf{train}})$, and the resulting per-node residuals r_i are written back into the graph state. This creates a closed feedback loop where the meta-learner observes the consequences of its updates in real time. Regarding the temporal horizon, while stochastic update depths are possible, we found that backpropagating through a fixed number of steps yielded the best results. We found that using longer values for T did yield better results, but that the computational cost of doing so was prohibitive. Because of this, and to be able to scale other hyperparameters such as model size and batch-size, we kept the BPTT truncation horizon relatively low at $T = 5$.

Outer Loop (Optimisation): After the T-step scan, the circuit loss $\mathcal{L}$ (Binary Cross Entropy between circuit output and target $\mathbf{Y}_{\mathbf{train}}$) is computed. We support both fixed evaluation at the final step ($t^* = T$) and stochastic evaluation, where t^* is drawn uniformly from $\{T_{\text{min}}, \ldots, T\}$

per sample. The latter could be used to get different "qualities" of gradients per batch (more immediate vs deeper in the recurrent graph), and enables the usage of schedulers controlling the number of steps used (start with short horizons, end with longer ones). Gradients of $\mathcal{L}$ with respect to the NCA parameters are computed via BPTT through the differentiable scan and circuit execution, and parameters of the NCA are updated using Adam (Kingma and Ba, 2014).

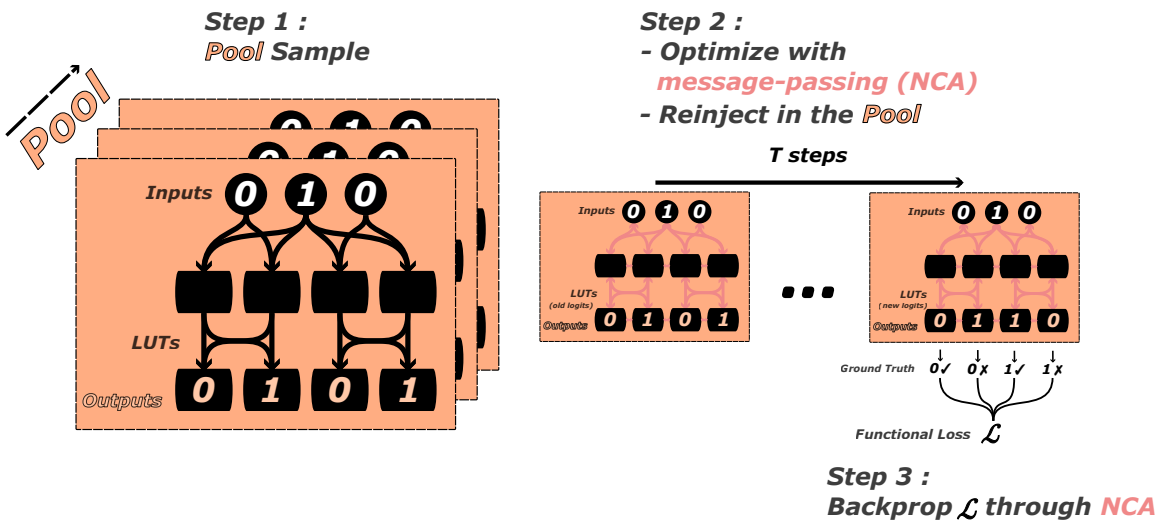


Figure 4.2: **Summary of the meta-learning training framework. (1) Sampling:** A batch of evolving circuit graphs is drawn from the persistent pool. **(2) Recurrent Rollout:** The sampled graphs are optimized for a fixed horizon T using the NCA message passing scheme. Crucially, functional execution occurs at every step to generate error feedback. **(3) Optimization:** The final logits are extracted, and the circuit is evaluated on the Boolean task. The resulting loss $\mathcal{L}$ is backpropagated to update the NCA policy, and the updated graphs are returned to the pool.

Experimental Setup: Our experiments, involving unrolling a recurrent transformer through multiple steps for batches of boolean input-output pairs and meta-batches of circuits, compounds in a significant computational cost. Most of our experiments can be run on a single GPU with enough VRAM (in our case NVIDIA RTX 4500, 24gb) thanks to optimization tricks and gradient checkpointing (jax.remat). Nonetheless, for the main sweeps of random seeds and regimes, we used the Imperial GPU cluster.

4.4 EXPERIMENTS AND RESULTS

To systematically evaluate the capabilities of the self-organising circuit framework, we designed a curriculum that progresses from established baselines to novel capabilities. Our experimental strategy unfolds in three phases:

1. **Regime I (Fixed Topologies):** We first validate the approach by mirroring the foundational questions of the original NCA paper—Growth, Persistence, and Repair—on a fixed circuit architecture.

2. **Regime II (Random Topologies):** We then increase the difficulty by removing the fixed architectural constraint, challenging the meta-learner to develop a generalist, wiring-agnostic routing policy.
3. **Regime III (Scale-Free Generalization):** Finally, we investigate the ultimate promise of the local update rule: the ability to train on small circuits and deploy the learned policy on significantly larger architectures.

4.4.1 *Regime I: Validation on Fixed Topologies*

In this first regime, we act strictly within the "NCA paradigm" (Mordvintsev et al., 2020), verifying that our Graph-NCA translation satisfies the core properties of self-organising systems: Growth, Persistence, and Repair.

Growth and Persistence

Our initial experiment validates the core premise: replacing global backpropagation with a local, recurrent update rule. In this regime, the Graph Pool is initialised with circuits sharing a single, fixed wiring architecture. The circuits start as "soft wires" (identity gates) with noisy logits, and the meta-optimiser is tasked with guiding them to a functional solution for a specific Boolean task. Unlike the original NCA work, which treated growth and persistence as separate training phases, we address both simultaneously. By employing the persistent pool mechanism from the outset, we force the policy to learn a solution that is not only functional at step T, but stable indefinitely. Crucially, to confirm that the system has learned the underlying Boolean function rather than merely memorising the training patterns, all reported metrics are computed strictly on the **held-out test split** (256 unseen input examples, see section 4.3.1). High accuracy on this split is the definitive signature of functional learning.

Results for the ideal execution are displayed in the **Top Row, Blue elements** of fig. 4.3. We observe that the NCA policy successfully converges across all three tasks, achieving performance virtually identical to the global Backpropagation baseline (faint blue bars):

- **Bit Reversal:** The NCA achieves perfect accuracy (1.0), matching the BP baseline and proving it can solve purely structural routing tasks.
- **Arithmetic Tasks:** For **Split Addition** and **Multiplication**, the NCA achieves mean accuracies of 0.96 and 0.84 respectively. Crucially, these distributions track the BP baseline closely, confirming that the decentralized update rule is capable of navigating the optimization landscape as effectively as a global gradient method in this fixed setting.

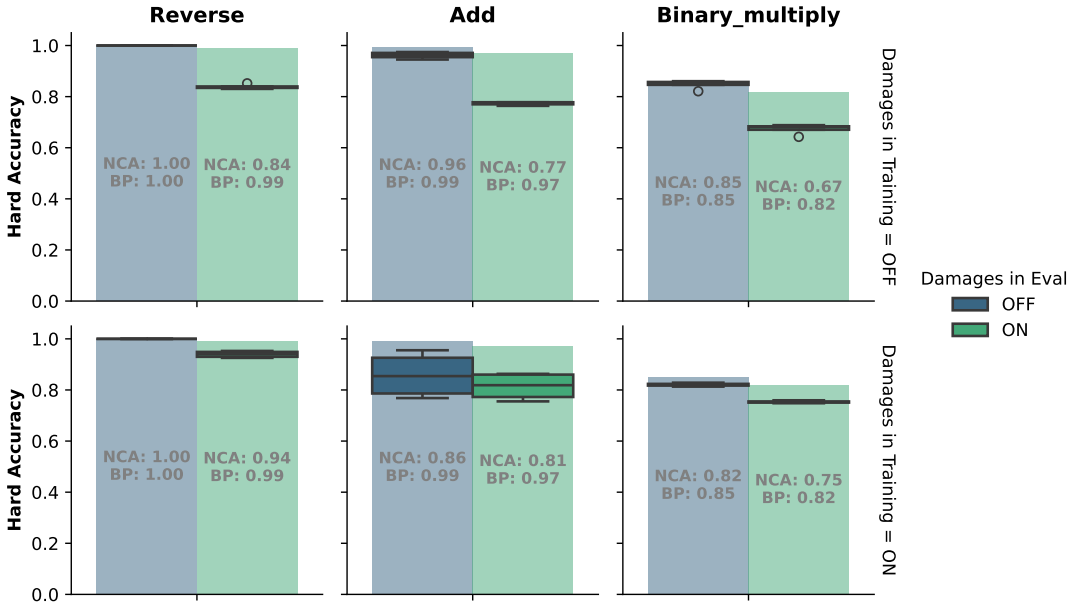


Figure 4.3: **NCA Performance on Fixed Topologies.** The **Box Plots** represent the test-set accuracy distribution of circuits grown by the NCA policy after 256 steps. The **Faint Background Bars** represent the baseline performance of standard Global Backpropagation (BP) on the same task and regime. The **Blue** elements (Damages OFF during Eval) show performance in ideal conditions. The **Green** elements (Damages ON during Eval) show performance when subjected to stochastic failures. We compare two training regimes: The **Top Row** displays a policy that has never faced damage during training (Damage Training = False); thus, the Green elements represent an out-of-distribution evaluation. The **Bottom Row** displays a policy trained with active damages (Damage Training = True), demonstrating adaptive robustness.

Robustness and Adaptive Resilience

Having established that the system can grow and persist, we explicitly target the third question: *Repair*. To counter hardware faults, we introduce two categories of damage: **Recoverable Damage** (transient soft errors) and **Permanent Damage** (stuck-at faults). This creates a "survival of the fittest" pressure, where the policy must solve the task while anticipating resource loss. We parametrize the damage system so that **20%** of the total number of gates will have failed by the time a circuit is reset (either in training, or through a single evaluation).

We analyse two distinct training regimes in fig. 4.3 to understand the emergence of robustness:

1. **Zero-Shot Resilience (Top Row):** Here, damage is never encountered during training. Evaluating with applied damages (Green box plots) is a pure out-of-distribution (OOD) task. We see that while the NCA takes a significant "hit" compared to the clean baseline, it retains partial functionality, suggesting an inherent robustness in the distributed representation even without explicit training.

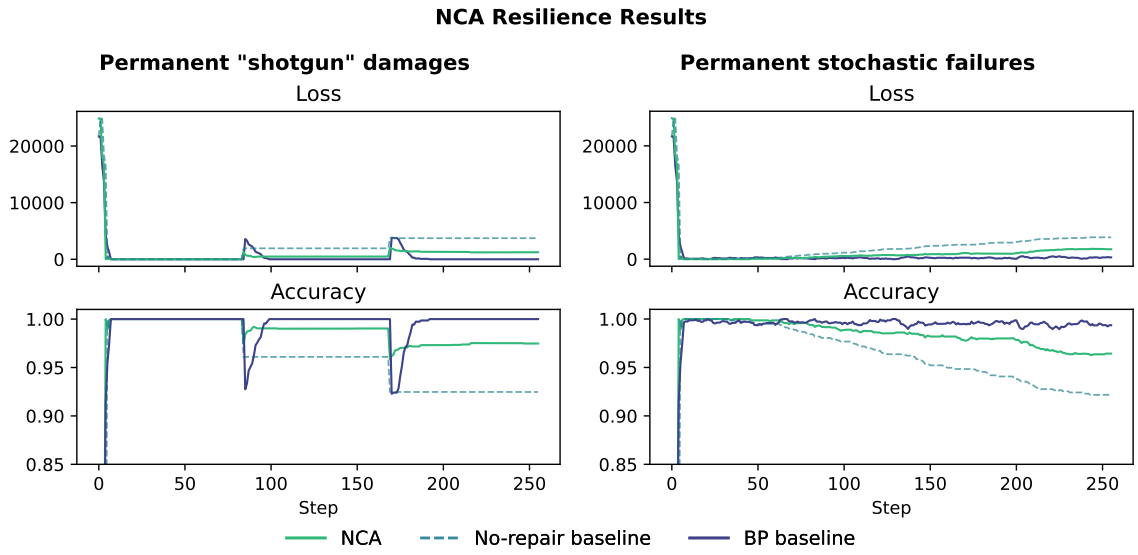


Figure 4.4: **Resilience to Damage.** Evolution of Loss (left) and Accuracy (right) under permanent damage. The **Solid Green** line represents the NCA-maintained circuit. The **Solid Blue** line represents standard Backpropagation (Upper Bound). The **Dashed** line represents the No-Repair baseline (Lower Bound).

2. **Learned Resilience (Bottom Row):** Here, we actively introduce damages in the training phase (mimicking the original NCA paper where images were cropped and re-grown).

In the robust training regime (Bottom Row), the damage "hit" is significantly reduced, showing that the NCA has adapted. In **Split Addition** and **Multiplication**, the performance when facing damages (Green) approaches the Backpropagation baseline (faint green bars), achieving parity or near-parity.

However, we note a trade-off: there is a slight degradation of the ceiling "no damage" performance (Bottom Row, Blue vs. Top Row, Blue). This suggests that the heavy damage rate during training (20%) adds substantial noise, slightly hindering the policy's ability to reach the perfect convergence seen in the clean regime.

To better understand the dynamics of this resilience, we place the NCA in damage regimes never seen during training (OOD), and analyse the circuit's response to damage over time against two baselines:

1. **Lower Bound (No-Repair):** A "shadow" network that stops receiving updates the moment damage begins.
2. **Upper Bound (BP):** A circuit optimized by standard Backpropagation (Adam) at every step.

As shown in fig. 4.4 (left), we apply a "shotgun" style catastrophic event where **10%** of the gates are instantly destroyed in a single step. This volley is repeated twice (totalling at 20%

destruction, matching the training total through a very different distribution). We also place the NCA in an unseen "delayed stochastic damage" regime (fig. 4.4, right), where the individual gate failures only start after an initial offset, to be able to compare it to an already functional lower-bound.

- **Vs. No-Repair (Lower Bound):** The NCA significantly outperforms the passive baseline. In the shotgun scenario, the passive circuit collapses. The NCA (green line) resists this drop and actively recovers, stabilizing at ≈ 0.97. This confirms the emergence of active self-repair.
- **Vs. Global BP (Upper Bound):** We observe that the decentralized NCA under-performs compared to Global Backpropagation (blue line), with the BP baseline recovering to almost perfect accuracy after damage. Nonetheless, the initial "hit" seems harder on the blind optimisation of BP, suggesting the NCA learned not only adaptative resilience (partial recovery) but also redundancy (minimizing initial impact).

Trajectories Analysis

To investigate the mechanism of this resilience, we analyse the optimization trajectories of the circuit logits in Principal Component (PC) space. We run a single NCA policy to optimize a batch of circuits under three conditions: no damage, recoverable damage, and permanent damage.

As shown in fig. 4.5, when no damage is applied (Left), every circuit follows an identical trajectory (we purposefully start with identical no-ops logits, witout random noise, to confirm this). When damage is introduced, the trajectories fan out. Crucially, this divergence occurs even under *recoverable* damage (Top Row), where the gates could theoretically be flipped back to their original state. This suggests that the NCA does not simply perform "error correction" to return to a memorised canonical state. Instead, it adapts dynamically: a temporary error pushes the system onto a new trajectory, and the policy guides it toward a *functionally equivalent but structurally distinct* local minimum. The system "re-routes" rather than "rewinds."

4.4.2 *Regime II: Generalisation to Random Topologies*

The most challenging setting removes the architectural constraints entirely. We transition to a **Random Topology** regime, where every circuit in the pool possesses a unique, randomly generated wiring diagram. Consequently, the meta-learner cannot overfit to "Gate A connected to Gate B." Instead, it must learn a truly topological, wiring-agnostic policy.

We find this learning regime to be significantly harder than Fixed Topology training. We successfully learned a high-performing, topology-agnostic policy for the **Bit Reversal** task,

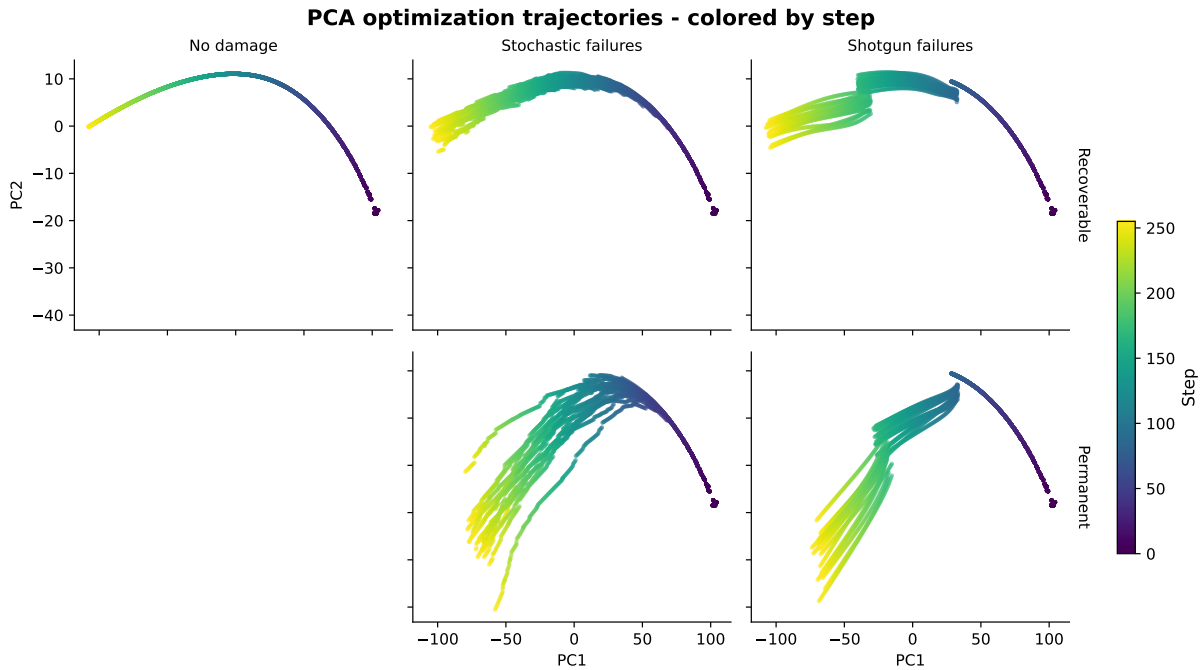


Figure 4.5: **PCA Trajectories of Circuit Optimization. (Left)** Without damage, the optimization is deterministic, following a single arc. **(Middle/Right)** Under damage, trajectories diverge significantly. Even with *recoverable* damage (top row), where the system could theoretically return to the original state, the policy instead drives the circuits into a diverse fan of alternative configurations.

achieving good generalization to unseen random graphs. However, for the arithmetic tasks (Addition and Multiplication), the policy struggles to fully converge. While it learns to approximate the output distribution—capturing coarse statistical patterns of the target function—it lacks the precision required for exact arithmetic operations. In this specific regime, standard Backpropagation (which is inherently topology-agnostic) still holds the advantage. However, the success on the Bit Reversal task provides a proof-of-principle that the topology-masked Transformer *can* learn generalisable routing algorithms. We display random-wiring training results in fig. 4.6

4.4.3 *Regime III: Scale-Free Optimisation*

Finally, we leverage the decentralised nature of our architecture to explore its scaling capabilities. Theoretically, because the Graph Transformer shares weights across all nodes and operates on local neighbourhoods, the learned update rule should be independent of the circuit size. We investigate this "Scale-Free" hypothesis by deploying a trained NCA policy on circuits significantly larger (or smaller) than those seen during training.

The results, displayed in fig. 4.7b, reveal a critical insight: scale-freedom is not just an architectural feature, but rather a *learned capability*.

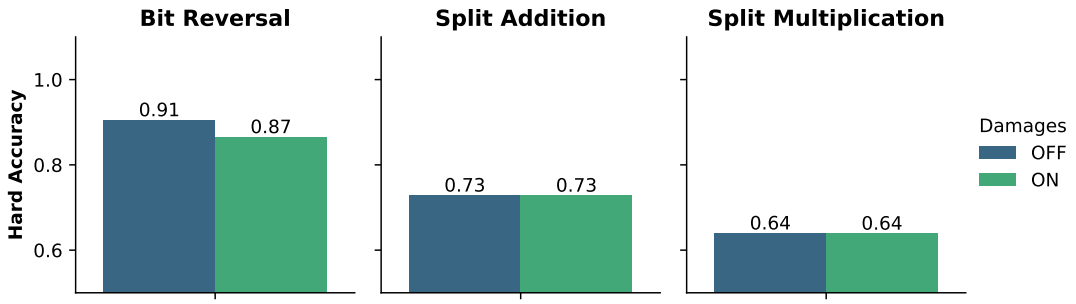


Figure 4.6: **NCA Performance on Random Topologies.** The **bars** represent the test-set accuracy distribution of circuits grown by the NCA policy after 256 steps. The **Blue** elements (Damages OFF) show performance in ideal conditions. The **Green** elements (Damages ON) show the performance of the *robust* policy when subjected to stochastic failures. BP performance is, by construction, wiring agnostic, and is not reported here (the accuracies of BP on each task are shown in fig. 4.3).

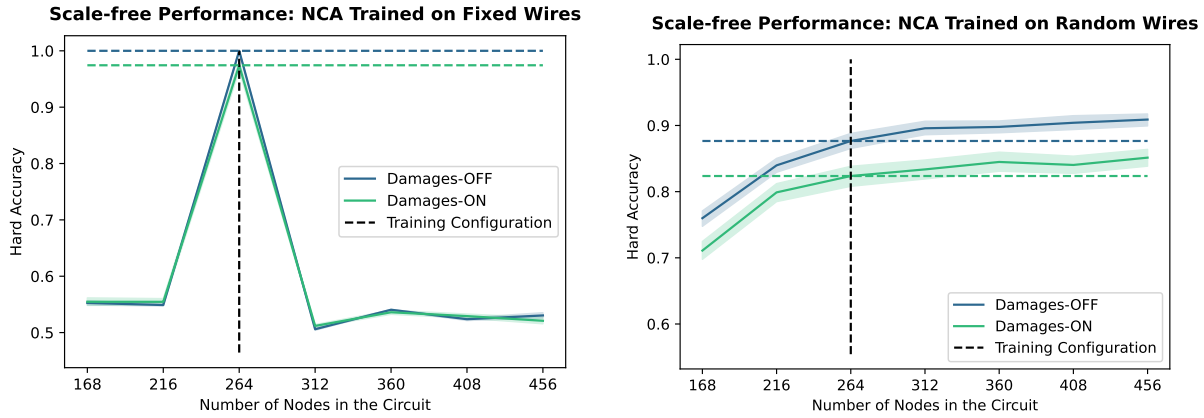
- **Overfitting Size (Fixed Training):** When the NCA is trained on a fixed topology (fig. 4.7a), it overfits the specific scale of the training graph. Performance peaks exactly at the training size ($N = 264$, vertical dashed line) and collapses for larger or smaller circuits.
- **Emergent Scalability (Random Training):** In contrast, when trained on the curriculum of random topologies (fig. 4.7b), the policy generalizes remarkably. Not only does it maintain function on larger circuits, but accuracy actually *increases* as we expand the width of the circuit (from 264 to 450+ nodes). The local policy is able to utilise the additional latent capacity of the wider layers to route signals more effectively, despite never having encountered graphs of this size during training (albeit the increase is marginal).

We note that this scaling success is currently limited to circuit *width*. Scaling *depth* remains a challenge, likely because our positional encoding (normalised depth fraction) changes resolution as layers are added, disrupting the policy's depth perception. Nevertheless, the ability to train on more small, cheap circuits and successfully deploy on wider architectures is a potent validation of the NCA's "growth" paradigm.

4.5 DISCUSSION

Summary

We have presented a framework for self-organising digital circuits that replaces global back-propagation with a decentralised, learned update rule. By extending the NCA paradigm from grid-based pattern formation to functional logic generation on arbitrary graphs, we demonstrated that a single topology-masked Transformer block, applied recurrently, can learn to configure Boolean circuits to satisfy computational tasks. On fixed topologies, the meta-learner assembles functional circuits from scratch and maintains homeostasis over extended time



(a) Trained on Fixed Wires

(b) Trained on Random Wires

Figure 4.7: **Scale-Free Generalization.** Test accuracy as a function of circuit width (number of nodes). The vertical dashed line indicates the circuit size used during training ($N = 264$). **(a)** When trained on a single fixed topology, the policy overfits the size: performance peaks at the training size and collapses elsewhere. **(b)** When trained on random topologies, the policy becomes truly scale-free: performance actually *improves* as the circuit gets wider, leveraging the extra capacity despite never seeing such large graphs during training.

horizons. When stochastic gate failures are introduced during training, the policy develops robustness—exhibiting both redundancy and active adaptation—that generalises to unseen, out-of-distribution fault patterns. Notably, repair does not consist of returning to a memorised pre-damage configuration; instead, the system navigates the solution manifold to discover *new* functional arrangements, consistent with the biological paradigm of adaptive plasticity. Extending to random topologies, we found that providing per-node error feedback (section 4.3.1) is essential for topology-agnostic optimisation, and the scale-free nature of the architecture enables deployment on circuits larger than those seen during training.

Limitations and Future Directions

Several limitations of the current framework point directly toward promising extensions. First, our positional encoding captures only vertical depth in the circuit DAG, offering no information about local graph topology. Incorporating *Random Walk Structural Encodings* (Dwivedi and Bresson, 2021)—which characterise local neighbourhood structure from the adjacency matrix alone—and edge-type annotations distinguishing forward logic wires from backward message-passing edges would provide richer structural context while preserving scale-freedom.

Second, the per-node error feedback r_i is a coarse signal: a single scalar residual per output gate, leaving the meta-learner blind to the actual input–output data. Augmenting the architecture with *cross-attention* to the task data, in the spirit of Perceiver IO (Jaegle et al., 2022), would enable genuine in-context reasoning—allowing the policy to inspect which inputs cause

which errors and make targeted, data-informed corrections rather than relying on aggregate residual signals. Preliminary experiments with this extension are ongoing and early results are encouraging.

Third, computational efficiency remains a bottleneck. The topology mask is currently applied as a dense $N \times N$ matrix, yet the actual connectivity is extremely sparse (under 3% density in our circuits). Since this sparsity is exact and known a priori from the wiring—fundamentally different from the approximate sparsity pursued in efficient Transformer research (Tay et al., 2022)—implementing native sparse attention would reduce both compute and memory to scale with the circuit's *wiring* rather than its *size*, making deployment on circuits with thousands of gates practical.

Together, these directions—richer structural encodings, direct data access, and sparse-native computation—point toward a meta-learner that is not only topology-agnostic and scale-free, but also task-aware and computationally efficient. Yet the deeper significance of this work may lie beyond the Boolean domain on which we have validated it. The framework we have presented—a shared, residual Transformer steering the functional states of nodes on a sparse graph—is a deliberately general mechanism, and its implications become most compelling when we consider replacing the logic gates with neurons... More on that in the General Conclusion !

GENERAL DISCUSSION AND CONCLUSION

Doing research, honestly, bravely and rigorously means constantly burying and mourning previous narratives, hypotheses and stories we used to tell ourselves, at the moment they inevitably show their limits. This PhD has been the story of such a journey. Dan once told me this: if you come up with what sounds like a very good idea on paper, and seemingly no one else has tried it (as far as you can tell), there are usually two possibilities:

THE MODULARITY TRAP

The first possibility is that it's not true.

Modularity is enticing. As we have seen throughout this manuscript, the concept has a powerful pull. We design modular systems, understand complex systems through their interacting parts, and feel like we observe this very notion all the way to some of the most complicated systems in our universe: biological brains. Modularity feels like a universal organising principle, and it aligns so naturally with how we think that we rarely question whether the map matches the territory.

But the more we looked for it, the harder it got to catch. Modularity is *elusive*. Chapter 1 showed us that even when we *impose* structural modularity—fixing the physical wiring into clean, separated clusters—functional specialization does not necessarily follow. The structure was there; the function refused to obey. Nevertheless, filled with courage and new ideas, we doubled down in Chapter 2, designing a purely compositional task with a strict separation of subroutines, seemingly perfect for a modular architecture, and moved away from rigid weight masks to let more organic forms of modularity emerge through spatial constraints. Even with all of this, functional specialization did not follow in the way we were expecting.

There is now a significant body of work actively deconstructing the myths and legends of modularity-land (see everything, everywhere, all at once ?), and it is clear that their work is essential. Theories such as neural reuse, that are testable, rigorous, yet significantly different in their postulates, offer precious alternatives to the entrenched status quo. A less modular, less localised view of intelligence may well be the more honest one, if we accept to pay the price that we understand it less than we thought. As is often the case in life, a crucial mission is to put forward constructive and exciting alternative views of thinking against the entrenched narrative (huh, are we still talking about science?).

Perhaps, then, the lesson of these first two chapters is not that modularity is absent, but that its causal threads are far messier than any tidy framework can capture. We designed our experiments precisely to isolate clean links—from pressure to signature, from structure to function—and what we found instead was contingency, context-dependence, and surprise. The framework from our introduction was useful for *organising* the question; but the answers, as always, refused to be neatly organised in return.

THE EMERGENCE PARADOX

The second possibility is that it's bloody hard to make it work.

Your ideas might be sound, your intuitions might be right, but the execution is just terribly difficult. That's why people tried before, and failed, and you haven't heard about it. Indeed, there is a lot that needs to go your way to validate hypotheses about modularity. Creating compositional tasks and datasets that might capture some structure of the world is hard. Grasping the right level of abstraction—detailed enough to be meaningful, simple enough to be tractable—is a constant balancing act. Controlling variables precisely enough to disentangle causal effects from confounds is harder still: this is most of what Chapter 2 was about. And even when control is achieved, the baselines and comparisons needed to make clean claims demand rigour and consistency. Here, perhaps, lies the most uncomfortable finding: the more rigorously we controlled our experiments—fair baselines, proper null hypotheses, clean variable isolation—the more the expected modularity seemed to dissolve. The methodology didn't fail us; it simply refused to confirm the story we wanted to tell.

But here lies a fundamental paradox: by gaining control, you relinquish reality. The more precisely you isolate a variable, the further you drift from the messy, entangled systems you set out to understand. And the converse is equally punishing: taking the multi-faceted nature of modularity seriously—accepting that it arises from the tension of many pressures, both physical and functional—places an enormous burden on experimental design. When you try to account for constraints, growth, function, composition, evolution, and spatial pressures simultaneously, the combinatorial complexity of what you need to account for explodes. You might as well say "let's model the whole universe and see what happens." A central skill of any good scientist is the ability to make the correct approximations: to abstract away just enough baggage to reveal the meaningful, without collapsing the magic.

There is also another, subtler trap—and this one, I must admit, caught me too. By voluntarily abstracting away from implementation details and discussing these concepts under a general "modularity" umbrella, we end up saying remarkably little about any actual system. A million ways exist to create neural modules, re-combine them, route them, engineer them, and they could all have a lot to teach us. The umbrella is so wide it might cover everything while

illuminating nothing. And perhaps this craving for a grand, unifying narrative—the desire to fit the messy diversity of intelligence under a single elegant concept—is itself a trap that our pattern-seeking brains set for us. Intelligence, and learning systems more broadly, may simply be too diverse for any single organising principle to capture.

This realisation, paradoxically, is also liberating. It led us to relinquish control over the precise form our systems would take, and to make everything "emerge" instead: structure through spatial embedding (Chapter 2), function through self-organisation (Chapters 3 and 4). But, we still did not relinquish the need to prescribe the outcome. And emergence has a way of messing with you—of finding surprising shortcuts and unexpected solutions that bear little resemblance to what you were hoping to see.

Yep, emergence is bloody hard.

Yet the struggles of the first half of this thesis were not wasted. They taught us something vital: not *what* modularity is, but *what it could need*. Constraints matter. Space matters. Locality matters. Resource pressure matters. These ingredients did not produce the clean, textbook modularity we were chasing, but they shaped the systems in interesting, unexpected ways. Perhaps the lesson is that the energy is better spent designing systems where interesting dynamics *can* happen, with fewer narrative biases about what those dynamics should be, than in trying to force a particular story to unfold. As they say about love: it might be when you look for it the least that you actually find it.

This is what the second half of the thesis pivoted toward. Chapter 3 stepped away from modularity entirely to ask a more fundamental question: can a self-organising substrate, governed by local rules, sustain general-purpose computation? And Chapter 4 turned those local rules into a learning rule—replacing global backpropagation with decentralised message passing on a functional circuit—and discovered that the resulting system was robust, adaptive, and scale-free. We stopped trying to *find* modularity and started building the kind of system where it *could happen* down the line.

TOWARD SELF-ORGANISING NEUROMORPHIC SUBSTRATES

This is where we get a potential third, tenuous, and not aforementioned possibility: it's a genuine, novel, and—if you're lucky—brilliant idea. Let's indulge for a bit. The following vision emerged largely through discussions with Maxence Faldor, my co-author on Chapter 3, who has been working separately on ideas similar in spirit to our self-organising digital circuits of Chapter 4, in the context of the brain.

The core postulate is this: *what if a single, shared local rule—operating on a sparse graph of heterogeneous units, under spatial and resource constraints—could both grow and teach a neural substrate?* What if development, learning, and repair were not separate engineering problems,

but different temporal expressions of the same underlying process? And what if everything we care about—robustness, specialisation, modularity—emerged as by-products?

In Chapter 4, we took a first step. We used an NCA to act as a learning rule on a functional substrate: Boolean digital circuits. But the framework we built—a shared, residual Transformer steering the functional states of nodes on a sparse graph—is deliberately more general than the Boolean domain on which we validated it. Its implications become most compelling when we consider what each element of this system maps onto, and what it could become.

Consider a pool of artificial neurons whose states, as in our circuit nodes, comprise both functional parameters and latent hidden channels. Edges on the graph represent physical synaptic connections, and the topology mask determines *who* can communicate with whom. The attention mechanism, operating within this scaffold, then determines *how* connected neurons influence one another. This framework provides a natural dissociation between two levels of organisation often conflated in artificial neural networks: the *structural connectivity*—the sparse physical wiring, governed by the mask—and the *functional interactions*—the dynamic, content-dependent coupling between linked neurons, governed by the learned attention policy. In our current system, the NCA operates entirely on the functional level: it steers node states while the topology remains fixed. But this clean separation immediately suggests a richer, dual-timescale architecture.

A developmental *growth* phase could leverage the NCA to lay down the structural scaffold itself: a spatially embedded topology shaped by the brain economy principle of Chapter 2, where long-range wiring is metabolically costly and modular, small-world organisation emerges without explicit priors. This phase could also initialise heterogeneous neuron properties: firing thresholds, response kernels, adaptation time constants: parameters that, much as in biological neurogenesis, are established during development and remain largely stable thereafter. A subsequent *learning* phase could then operate within this structural backbone, using the hidden state channels to mediate dynamic, context-dependent interactions between neurons. This is the fundamental departure from conventional artificial neural networks: rather than optimising fixed synaptic weights, the system maintains *living* functional couplings that fluctuate at every step (closer in spirit to short-term synaptic plasticity than to static weight matrices). The recurrent Transformer would thus serve a dual role: as a morphogenetic program that *grows* a network, and as a local learning rule that *teaches* it, both instantiated through the same shared-weight, residual update mechanism.

Crucially, because a single model governs both processes, the boundary between development and learning need not be a hard cutoff. In biological development, the bulk of the structural scaffold is established before birth yet continues to be sculpted by experience: through aggressive synaptic pruning in infancy and activity-dependent rewiring throughout life. Similarly, growth and learning here could operate on two overlapping *temporal scales* rather than in

discrete phases, each continuously informing the other: functional feedback shaping structural refinement, and structural reorganisation opening new functional possibilities.

Realising the structural half of this vision—endowing the NCA with the ability to add neurons, prune connections, and dynamically reshape the topology—remains the central open challenge. Yet it aligns naturally with the push toward sparse, efficient attention mechanisms! The biological sparsity of neural connectivity, in this case, would not be an obstacle to overcome but a computational advantage to exploit: each structural modification updates only a local patch of the attention mask, preserving scale-freedom, while making the entire system faster and more efficient. Space, as we have argued throughout this thesis, is not merely a passive constraint but an active *computational resource*: it introduces propagation delays that enable temporal coding, and—through the relentless pressure of wiring cost—sculpts the clustered, structurally-rich architectures that could underpin compositional intelligence.

And here, perhaps, the circle begins to close, partially at least. The introduction of this thesis proposed a framework for understanding modularity through a grid of Pressures and Signatures, Proximate and Ultimate (Table 0.1). Chapters 1 and 2 explored this grid methodically, isolating quadrants, testing causal links between them. External pressures—energy constraints, task compositionality—will always remain outside the system; no self-organising substrate generates its own physics or its own survival demands. But *within* the system, something genuinely new happens: the structural output of the growth phase (a Proximate Signature) becomes the constraint under which learning operates (a Proximate Pressure for the functional phase), and functional feedback can in turn reshape the scaffold. The grid, at least along this internal axis, becomes a loop. This is, in a sense, the Baldwin Effect that the introduction described (section 0.6): learned signatures retro-acting on the developmental scaffold, blurring the line between what is innate and what is acquired. When a single model governs both development and learning, this retroaction is not a metaphor—it is the literal mechanism.

What might emerge from such a system? A spatially embedded, resource-constrained pool of heterogeneous neurons, governed entirely by a shared local rule that mediates development, learning, and repair... This sounds like, in essence, a minimal *in silico* model of a self-organising brain. Could travelling waves of coordinated activity arise spontaneously from the interplay of local dynamics and spatial structure? Could functional hierarchies and specialisation develop without top-down design? Could modularity itself—that elusive property we chased across four chapters—finally emerge, not because we engineered it, but because we created the conditions for it?

Of course, one might observe the irony: after four chapters of demonstrating that modularity is elusive, that emergence resists our expectations, and that grand frameworks have a way of flattering our narrative instincts, I am ending this thesis by arguing that with the *right* framework, *this time*, it will all come together. The pull of the modular narrative is strong—perhaps it got me one last time.

But even if it did, I believe the principles themselves have earned their keep. Locality, adaptive resilience, scale-freedom, spatial constraint, and the primacy of self-organisation: these are not abstractions waiting for modularity to validate them. They were demonstrated, concretely, across the chapters of this thesis. And the intelligence they could point toward—one that grows, learns, and heals itself from the ground up—is worth building regardless of whether our favourite emergent property finally decides to show up.

Montpellier, February 13, 2026

A

APPENDIX FOR CHAPTER 1

A.1 Q MEASURE DERIVATION

Newman (2006) defines the modularity measure Q for an undirected graph. We modify the definition for a directed graph as:

$$Q = \frac{1}{M}\sum_{ij}(A_{ij} - P_{ij})\delta_{g_i g_j} = \frac{1}{M}\sum_{ij}\left(A_{ij} - \frac{k_i^{out}k_j^{in}}{M}\right)\delta_{g_i g_j} \tag{A.1}$$

where M is the total number of edges in the network, A is the adjacency matrix, δ is the Kronecker delta, g_i is the group index of node i, k_i^{out}/k_j^{in} are the out-degree/in-degree of node i/j respectively, used to compute P_{ij}: the probability of a connection between nodes i and j if we randomized the edges respecting node degrees. For our network, we have 2

sub-networks (with group index 0 and 1 respectively) of n neurons, densely connected, with a fraction p of active inter-connections between the two. From this we get:

$$M = 2n^2(1+p)$$

$$\forall i \in [1, 2n], \quad g_i = \begin{cases} 0, & \text{if } i \in [0, n-1] \\ 1, & \text{if } i \in [n, 2n] \end{cases}$$

$$\forall (i,j) \in [0, 2n]^2, \quad A_{i,j} = \begin{cases} 1, & \text{if } \delta_{g_i g_j} = 1 \\ 0 \text{ or } 1, & \text{if } \delta_{g_i g_j} = 0 \end{cases}$$

$$\forall (i,j) \in [0, 2N]^2, \quad \begin{cases} k_i^{out} = \sum_{j'} A_{ij'} \approx n(1+p) \\ k_j^{in} = \sum_{i'} A_{i'j} \approx n(1+p) \end{cases}$$

$$\text{And} \quad \begin{cases} \sum_i k_i^{out} = \sum_i \sum_{j'} A_{ij'} = n^2(1+p) \\ \sum_j k_j^{in} = \sum_j \sum_{i'} A_{i'j} = n^2(1+p) \end{cases}$$

$$\text{Thus: } Q = \frac{1}{2n^2(1+p)} \sum_{ij} \left(A_{ij} - \frac{(1+p)}{2} \right) \delta_{g_i g_j}$$

$$Q = \frac{1}{2n^2(1+p)} (2n^2) \left(1 - \frac{(1+p)}{2} \right)$$

$$\boxed{Q = \frac{1-p}{2(1+p)}}$$

A.2 NETWORK ACCURACY

fig. A.1 displays the network's accuracy for results 3.1. We always test on a separate testing set, and train for a fixed number of epochs to avoid overfitting, repeated for 10 random seeds. We can see that, when the sparsity of communications is very high, networks only manage to perform slightly above chance (~60%). This shows that the task is not trivial for networks of that size, which was one of our objectives so as to not over-parameterize the networks and make the results meaningful.

A.3 DECISION DYNAMICS

fig. A.2 showcases decision dynamics over the course of training, for more or less modular networks, demonstrating the point made in section 1.4.2. We see that the more interconnected,

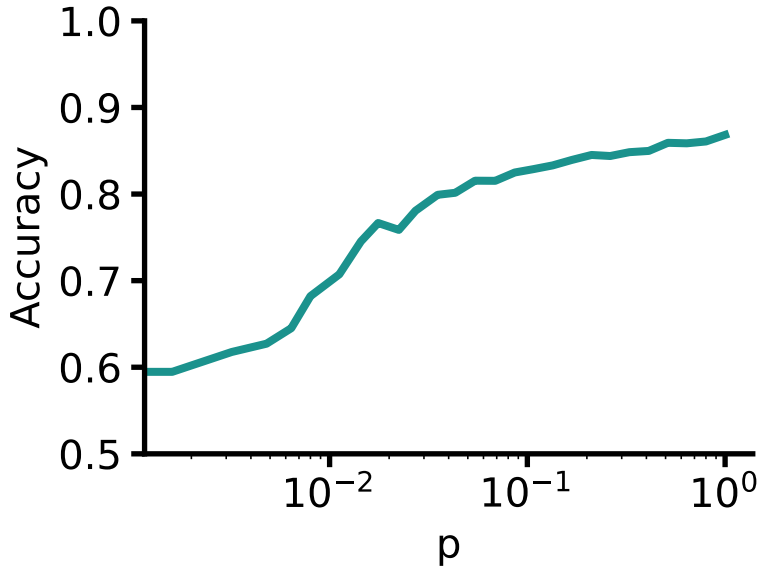


Figure A.1: Accuracy for networks composed of 25-neurons modules, with varying levels of inter-module communication p. Results are shown for 5 independent experiments per parameter-set (sparsity), with confidence intervals showed as shaded areas.

the more the networks tend to rely on a single decision-taking agent (decision going up to 1 or falling closer to 0 on average) Targets correspond to the two targets of eq. (1.2), and we can see that the small bias discussed when presenting the tasks is present at every sparsity, but getting ever more present in more interconnected networks.

A.4 PARAMETER SWEEPS FOR STRUCTURE-FUNCTION RELATIONSHIP

Figure A.3 shows the complete parameter sweeps referred to in section section 1.4.2.

A.5 CODE EXAMPLES

A.5.1 *Noisy inputs*

In section section 1.4.3, we investigate the different dynamics of functional specialization in networks unrolled in time. To that end, we add noisy dynamics to the input, which are by default identical at every time-steps, and create a noisy version of the dataset. For each training batch, and at each time-step, we add an average from a few other random samples of the same batch, multiplied by a noise-level. Example code is provided here This would have to be repeated at each time-step (see following section).

```
def add_structured_noise(batch_t, n_samples=5, noise_ratio=0.9):
```

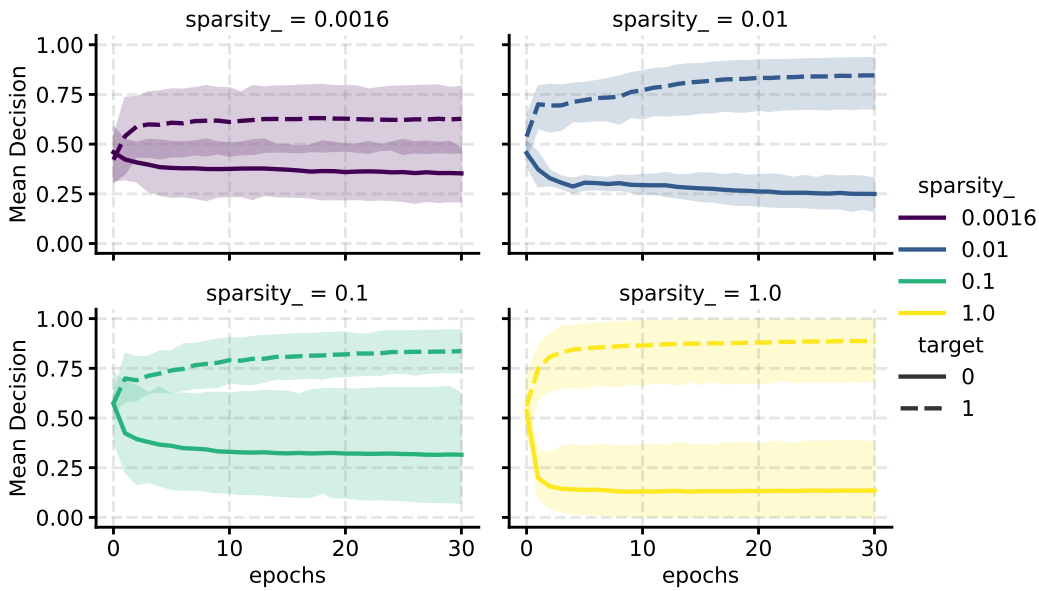


Figure A.2: Decision-making dynamics over training, for networks composed of 25 neurons per module. Lines represent the average decision making in networks (1 being always module 1 and 0 always module 0). Results are shown for 5 independent experiments per parameter-set (sparsity), with confidence intervals showed as shaded areas.

```
    noised_idxs = np.stack(
        [
            np.random.choice(batch_t.shape[0], size=n_samples, replace=False)
            for _ in range(batch_t.shape[0])
        ]
    )
    noisy_samples = batch_t[noised_idxs] * (
        np.random.rand([n_samples] + list(batch_t.shape[1:])) < (1 / n_samples)
    )
    noisy_batch = (1 - noise_ratio) * batch_t + noise_ratio * noisy_samples.mean(1)
    return noisy_batch
```

A.5.2 *Dynamic inputs*

Here is how we create input with noisy dynamics, and potentially stochastic start times.

```
def create_stochastic_batch(batch, n_samples, noise_ratio, random_start) :
    noisy_batch = np.stack(
        [add_structured_noise(batch_t, n_samples, noise_ratio) for batch_t in batch]
    )
    if random_start :
```

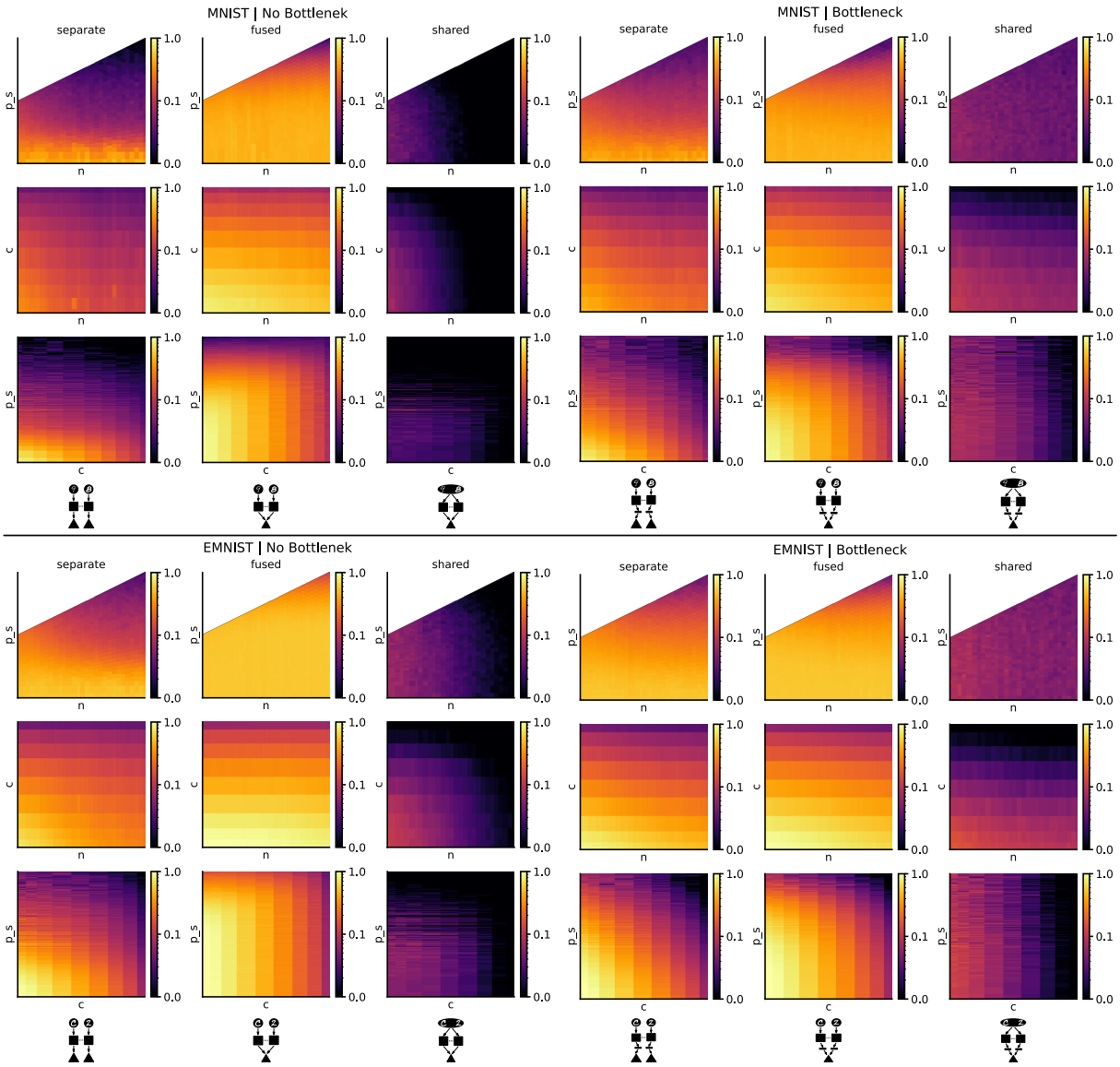


Figure A.3: Specialization for all architectures and data. See caption to figure fig. 1.3.

```
start_times = np.random.randint(1, batch.shape[0] - 1, (batch.shape[1],))
mask = (
            np.arange(nb_steps)[:, None]
            >= start_times[None, :,]
        )
 pure_noise = np.stack(
    [add_structured_noise(batch_t, n_samples, 1.0) for batch_t in batch]
)

stochastic_timing_batch = noisy_batch * mask + pure_noise * (not mask)
return stochastic_timing_batch
```

```
    else :
        return noisy_batch
```

B

APPENDIX FOR CHAPTER 2

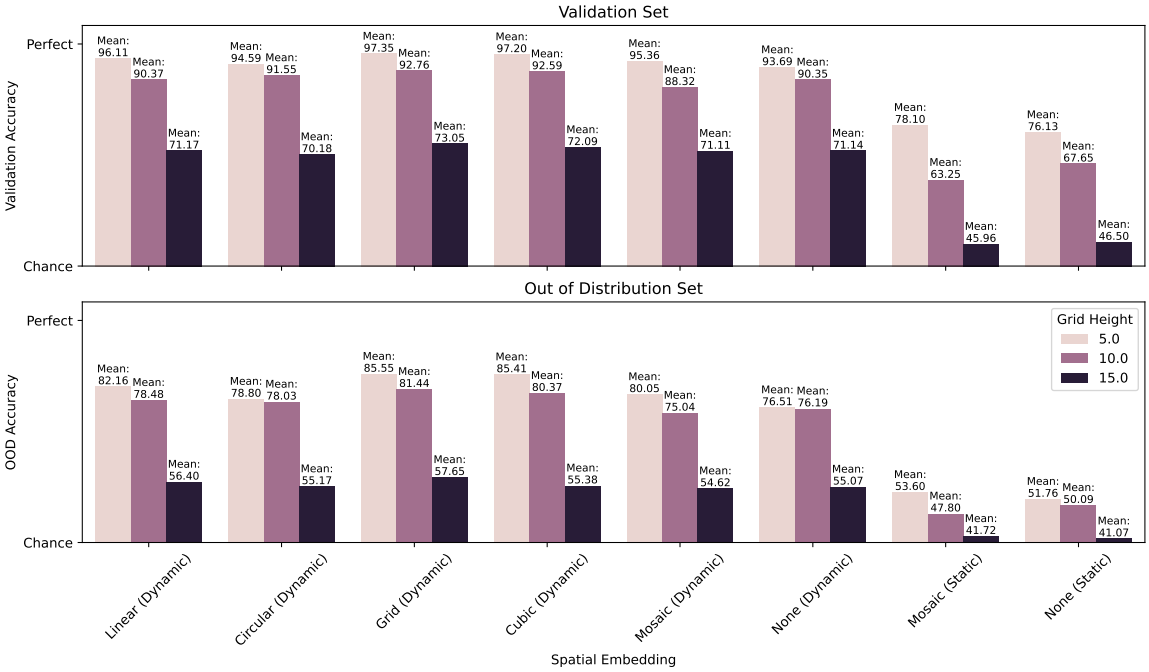


Figure B.1: General Accuracy results of models, on the validation and OOD sets, displayed for multiple grid heights values.

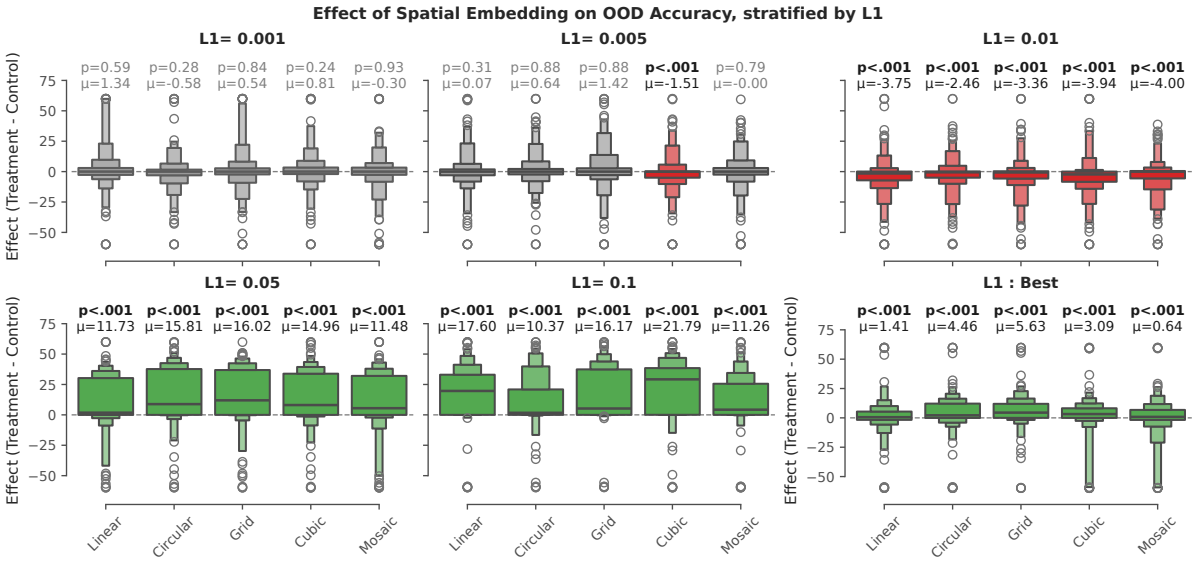


Figure B.2: Effect of spatial embedding on OOD accuracy, stratified by average L1 strength applied.

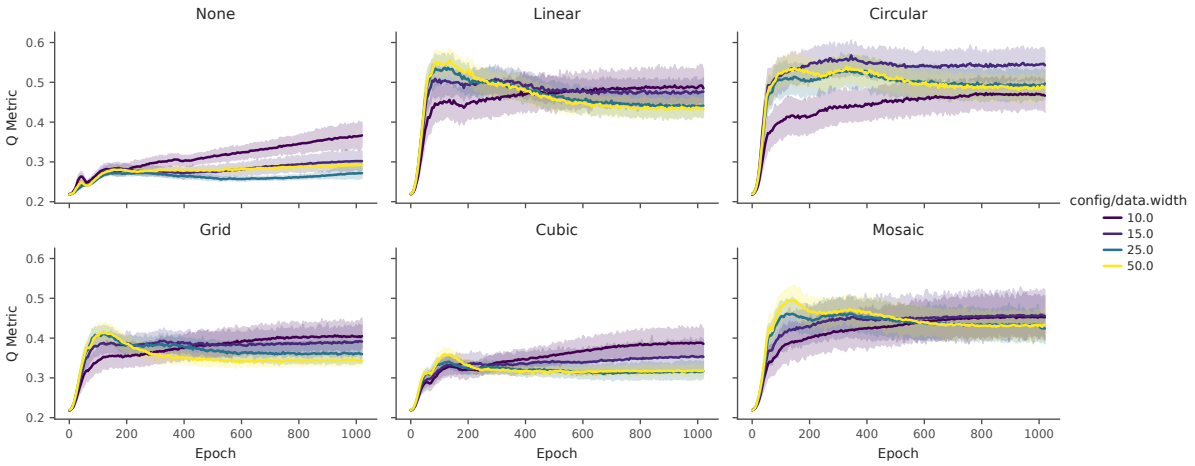


Figure B.3: Average evolution over training of the Q metric computed on weighted recurrent matrices, using the louvain community detection algorithm, for different spatial embeddings and dataset widths. We only plot runs with L1 regularization that led to the best performance, everything else being equal, removing high L1 runs that'd have led to high Q but poor accuracy.

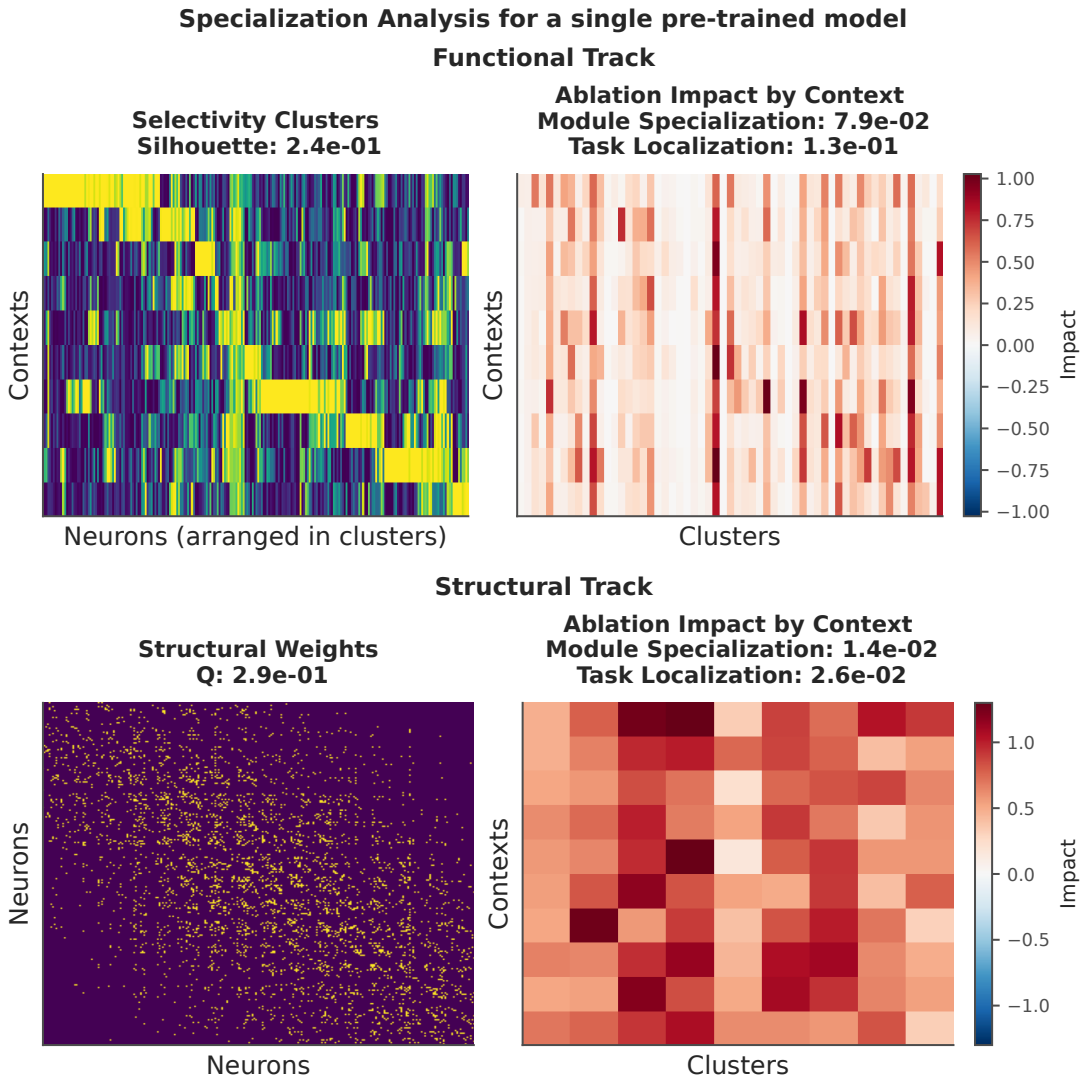


Figure B.4: An example of the specialization analysis applied to a trained model (grid embedding)

AUTHOR'S LIST OF PUBLICATIONS

Gabriel Béna and Dan F. M. Goodman. Dynamics of specialization in neural modules under resource constraints. *Nature Communications*, 16(1):187, January 2025. ISSN 2041-1723. doi: 10.1038/s41467-024-55188-9. URL https://www.nature.com/articles/s41467-024-55188-9. Publisher: Nature Publishing Group. (page 43).

Gabriel Béna, Timo Wunderlich, Mahmoud Akl, Bernhard Vogginger, Christian Mayr, and Hector Andres Gonzalez. Event-based backpropagation on the neuromorphic platform SpiNNaker2, March 2025a. URL http://arxiv.org/abs/2412.15021. arXiv:2412.15021 [cs].

Gabriel Béna, Maxence Faldor, Dan F. M. Goodman, and Antoine Cully. A Path to Universal Neural Cellular Automata, May 2025b. URL http://arxiv.org/abs/2505.13058. arXiv:2505.13058 [cs]. (page 93).

Marcus Ghosh, **Gabriel Béna**, Volker Bormuth, and Dan F. M. Goodman. Nonlinear fusion is optimal for a wide class of multisensory tasks. *PLOS Computational Biology*, 20(7):e1012246, July 2024a. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1012246. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1012246. Publisher: Public Library of Science.

Marcus Ghosh, Karim G. Habashy, Francesco De Santis, Tomas Fiers, Dilay Fidan Erçelik, Balázs Mészáros, Zachary Friedenberger, **Gabriel Béna**, Mingxuan Hong, Umar Abubacar, Rory T. Byrne, Juan Luis Riquelme, Yuhan Helena Liu, Ido Aizenbud, Brendan A. Bicknell, Volker Bormuth, Alberto Antonietti, and Dan F. M. Goodman. Spiking neural network models of sound localisation via a massively collaborative process, July 2024b. URL https://www.biorxiv.org/content/10.1101/2024.07.19.604252v1.

Vlad C. Andrei, Alexandru P. Drăgutoiu, **Gabriel Béna**, Mahmoud Akl, Yin Li, Matthias Lohrmann, Ullrich J. Mönich, and Holger Boche. Deep- Unrolling Multidimensional Harmonic Retrieval Algorithms on Neuromorphic Hardware. In *2024 58th Asilomar Conference on Signals, Systems, and Computers*, pages 298–302, October 2024. doi: 10.1109/IEEECONF60004.2024.10942794. URL https://ieeexplore.ieee.org/abstract/document/10942794. ISSN: 2576-2303.

Gabriel Béna and Dan F. M. Goodman. Extreme sparsity gives rise to functional specialization. *arXiv:2106.02626 [cs, q-bio]*, 2021. URL http://arxiv.org/abs/2106.02626v1. arXiv:2106.02626. (page 43).

BIBLIOGRAPHY

Abnar, S., Saremi, O., Dinh, L., Wilson, S., Bautista, M. A., Huang, C., Thilak, V., Littwin, E., Gu, J., Susskind, J. and Bengio, S. (2023), 'Adaptivity and Modularity for Efficient Generalization Over Task Complexity'. arXiv:2310.08866. http://arxiv.org/abs/2310.08866 (accessed on: 2024-12-02) (pages 67, 68, 69, 71, 73).

Achard, S. and Bullmore, E. (2007), 'Efficiency and cost of economical brain functional networks', *PLoS computational biology* **3**(2), e17. doi: 10.1371/journal.pcbi.0030017 (page 45).

Achterberg, J., Akarca, D., Strouse, D. J., Duncan, J. and Astle, D. E. (2023), 'Spatially embedded recurrent neural networks reveal widespread links between structural and functional neuroscience findings', *Nature Machine Intelligence* **5**(12), 1369–1381. Publisher: Nature Publishing Group. doi: 10.1038/s42256-023-00748-9 (pages 36, 60, 69, 77, 88).

ADA University, Gladkikh, V., Nigay, A. and International University of Information Technologies (2018), 'Wireworld++: A cellular automaton for simulation of nonplanar digital electronic circuits', *Complex Systems* **27**(1), 19–44. doi: 10.25088/ComplexSystems.27.1.19 (page 96).

Adolphs, R. (2016), 'Human Lesion Studies in the 21st Century', *Neuron* **90**(6), 1151–1153. Publisher: Elsevier. doi: 10.1016/j.neuron.2016.05.014 (page 46).

Adolphs, R. and Anderson, D. J. (2018), *The neuroscience of emotion: a new synthesis*, Princeton University Press, Princeton. OCLC: on1004927099. (page 46).

Agüera y Arcas, B. (2025), *What is intelligence? lessons from AI about evolution, computing, and minds*, Antikythera, The MIT Press, Cambridge, Massachusetts London, England. (page 38).

Ali, A., Ahmad, N., de Groot, E., Johannes van Gerven, M. A. and Kietzmann, T. C. (2022), 'Predictive coding is a consequence of energy efficiency in recurrent neural networks', *Patterns* **3**(12), 100639. doi: 10.1016/j.patter.2022.100639 (page 61).

Amer, M. and Maul, T. (2019), 'A Review of Modularization Techniques in Artificial Neural Networks', *Artificial Intelligence Review* **52**(1), 527–561. arXiv:1904.12770 [cs]. doi: 10.1007/s10462-019-09706-7 (page 32).

Anderson, M. L. (2010), 'Neural reuse: A fundamental organizational principle of the brain', *Behavioral and Brain Sciences* **33**(4), 245–266. doi: 10.1017/S0140525X10000853 (page 29).

Anderson, M. L. (2014), *After phrenology: neural reuse and the interactive brain*, The MIT Press, Cambridge, Massachusetts ; London, England. (page 29).

Anderson, M. L., Kinnison, J. and Pessoa, L. (2013), 'Describing functional diversity of brain regions and brain networks', *NeuroImage* **73**, 50–58. doi: 10.1016/j.neuroimage.2013.01.071 (page 29).

Andreas, J., Rohrbach, M., Darrell, T. and Klein, D. (2017), 'Neural Module Networks', *arXiv:1511.02799 [cs]* . arXiv: 1511.02799. Retrieved from http://arxiv.org/abs/1511.02799 (page 68).

Ba, J. L., Kiros, J. R. and Hinton, G. E. (2016), 'Layer normalization'. https://arxiv.org/abs/1607.06450 (page 120).

Bachlechner, T., Majumder, B. P., Mao, H. H., Cottrell, G. W. and McAuley, J. (2020), 'ReZero is All You Need: Fast Convergence at Large Depth'. arXiv:2003.04887 [cs]. http://arxiv.org/abs/2003.04887 (accessed on: 2025-06-18) (page 121).

Backus, J. (2007), Can programming be liberated from the von neumann style? a functional style and its algebra of programs, *in* 'ACM Turing Award Lectures', Association for Computing Machinery, p. 1977. doi: 10.1145/1283920.1283933 (page 97).

Bahdanau, D., Murty, S., Noukhovitch, M., Nguyen, T. H., de Vries, H. and Courville, A. (2019), 'Systematic Generalization: What Is Required and Can It Be Learned?'. arXiv:1811.12889 [cs]. http://arxiv.org/abs/1811.12889 (accessed on: 2023-05-03) (pages 60, 61).

Bakhtiari, S., Mineault, P., Lillicrap, T., Pack, C. and Richards, B. (2021), The functional specialization of visual cortex emerges from training parallel pathways with self-supervised predictive learning, *in* 'Advances in Neural Information Processing Systems', Vol. 34, Curran Associates, Inc., pp. 25164–25178. Retrieved from https://proceedings.neurips.cc/paper/2021/hash/d384dec9f5f7a64a36b5c8f03b8a6d92-Abstract.html (pages 35, 61).

Baldwin, C. Y. and Clark, K. B. (2000), *Design Rules, Volume 1: The Power of Modularity*, The MIT Press. doi: 10.7551/mitpress/2366.001.0001 (page 20).

Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., Pritzel, A., Chadwick, M. J., Degris, T., Modayil, J., Wayne, G., Soyer, H., Viola, F., Zhang, B., Goroshin, R., Rabinowitz, N., Pascanu, R., Beattie, C., Petersen, S., Sadik, A., Gaffney, S., King, H., Kavukcuoglu, K., Hassabis, D., Hadsell, R. and Kumaran, D. (2018), 'Vector-based navigation using grid-like representations in artificial agents', *Nature* **557**(7705), 429–433. Number: 7705 Publisher: Nature Publishing Group. doi: 10.1038/s41586-018-0102-6 (page 47).

Barrett, H. C. and Kurzban, R. (2006), 'Modularity in cognition: Framing the debate', *Psychological Review* **113**(3), 628–647. Place: US Publisher: American Psychological Association. doi: 10.1037/0033-295X.113.3.628 (page 25).

Bartolozzi, C., Indiveri, G. and Donati, E. (2022), 'Embodied neuromorphic intelligence', *Nature Communications* **13**(1), 1024. Number: 1 Publisher: Nature Publishing Group. doi: 10.1038/s41467-022-28487-2 (page 60).

Bassett, D. S. and Bullmore, E. (2006), 'Small-world brain networks', *The Neuroscientist* **12**(6), 512–523. doi: 10.1177/1073858406293182 (page 97).

Bassett, D. S., Greenfield, D. L., Meyer-Lindenberg, A., Weinberger, D. R., Moore, S. W. and Bullmore, E. T. (2010), 'Efficient Physical Embedding of Topologically Complex Information Processing Networks in Brains and Computer Circuits', *PLOS Computational Biology* **6**(4), e1000748. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1000748 (page 28).

Bassett, D. S. and Sporns, O. (2017), 'Network neuroscience', *Nature Neuroscience* **20**(3), 353–364. Number: 3 Publisher: Nature Publishing Group. doi: 10.1038/nn.4502 (pages 26, 44, 61).

Bassett, D. S., Wymbs, N. F., Porter, M. A., Mucha, P. J., Carlson, J. M. and Grafton, S. T. (2011), 'Dynamic reconfiguration of human brain networks during learning', *Proceedings of the National Academy of Sciences* **108**(18), 7641–7646. Publisher: Proceedings of the National Academy of Sciences. doi: 10.1073/pnas.1018985108 (page 58).

Bechtel, W. and Richardson, R. C. (1993), *Discovering complexity: Decomposition and localization as strategies in scientific research*, Discovering complexity: Decomposition and localization as strategies in scientific research, Princeton University Press, Princeton, NJ, US. Pages: xiv, 286. (pages 25, 29).

Beevor, C. E. and Horsley, V. A. H. (1887), 'VI. A minute analysis (experiments) of the various movements produced by stimulating in the monkey different regions of the cortical centre for the upper limb, as defined by Professor Ferrier', *Philosophical Transactions of the Royal Society of London. (B.)* (178), 153–167. doi: 10.1098/rstb.1887.0006 (page 24).

Bellec, G., Kappel, D., Maass, W. and Legenstein, R. (2018), 'Deep Rewiring: Training very sparse deep networks'. arXiv:1711.05136 [cs]. http://arxiv.org/abs/1711.05136 (accessed on: 2025-12-19) (pages 75, 76).

Bennett, C. (1995), 'Logical Depth and Physical Complexity', *Computerkultur* . doi: 10.1007/978-3-7091-6597-3_8 (page 37).

Betzel, R. F. (2020*a*), 'Community detection in network neuroscience'. arXiv:2011.06723 [q-bio]. http://arxiv.org/abs/2011.06723 (accessed on: 2023-04-14) (page 45).

Betzel, R. F. (2020*b*), 'Organizing principles of whole-brain functional connectivity in zebrafish larvae', *Network Neuroscience* **4**(1), 234–256. doi: 10.1162/netn_a_00121 (page 46).

Betzel, R. F., Avena-Koenigsberger, A., Goñi, J., He, Y., de Reus, M. A., Griffa, A., Vértes, P. E., Mišic, B., Thiran, J.-P., Hagmann, P., van den Heuvel, M., Zuo, X.-N., Bullmore, E. T. and Sporns, O. (2016), 'Generative models of the human connectome', *NeuroImage* **124**, 1054–1064. doi: 10.1016/j.neuroimage.2015.09.041 (page 28).

Blondel, V. D., Guillaume, J.-L., Lambiotte, R. and Lefebvre, E. (2008), 'Fast unfolding of communities in large networks', *Journal of Statistical Mechanics: Theory and Experiment* **2008**(10), P10008. doi: 10.1088/1742-5468/2008/10/P10008 (page 82).

Blumer, A., Ehrenfeucht, A., Haussler, D. and Warmuth, M. K. (1989), 'Learnability and the Vapnik-Chervonenkis dimension', *J. ACM* **36**(4), 929–965. doi: 10.1145/76359.76371 (page 66).

Boeshertz, G. and Clopath, C. (2025), 'Predictive learning enables compositional representations'. ISSN: 2692-8205 Pages: 2025.09.26.678731 Section: New Results. https://www.biorxiv.org/content/10.1101/2025.09.26.678731v1 (accessed on: 2025-11-16) (pages 35, 81).

Broca, M. P. (1861), 'REMARQUES SUR LE SIÉGE DE LA FACULTÉ DU LANGAGE ARTICULÉ, SUIVIES D'UNE OBSERVATION D'APHÉMIE (PERTE DE LA PAROLE)'. (page 23).

Brodmann, K. (1909), *Vergleichende Lokalisationslehre der Grosshirnrinde in ihren Prinzipien dargestellt auf Grund des Zellenbaues*, Barth. (pages 24, 44).

Bullmore, E. and Sporns, O. (2012*a*), 'The economy of brain network organization', *Nature Reviews Neuroscience* **13**(5), 336–349. Publisher: Nature Publishing Group. doi: 10.1038/nrn3214 (pages 28, 45, 69).

Bullmore, E. T. and Sporns, O. (2012*b*), 'The economy of brain network organization', *Nature Reviews Neuroscience* **13**, 336–349. (page 97).

Carter, W., Duong, I., Freman, R., Hsieh, H., Ja, J., Mahoney, J., Ngo, N. and Sac, S. L. (1986), A user programmable reconfigurable logic array, *in* 'Proceedings of the IEEE Custom Integrated Circuits Conference'. Retrieved from https://www.semanticscholar.org/paper/A-user-programmable-reconfigurable-logic-array-Carter-Duong/0f7d6857d51b096d44dc59ccfecb8e0d2468d936 (page 115).

Casper, S., Hod, S., Filan, D., Wild, C., Critch, A. and Russell, S. (2022), Graphical Clusterability and Local Specialization in Deep Neural Networks. Retrieved from https://openreview.net/forum?id=HreeeJvkue9 (pages 45, 61).

Chan, B. W.-C. (2019), 'Lenia - biology of artificial life', *Complex Systems* **28**(3), 251–286. doi: 10.25088/ComplexSystems.28.3.251 (pages 95, 97, 115).

Chen, G., Kang, B., Lindsey, J., Druckmann, S. and Li, N. (2021), 'Modularity and robustness of frontal cortical networks', *Cell* **184**(14), 3717–3730.e24. doi: 10.1016/j.cell.2021.05.026 (page 44).

Chen, X., Liang, C., Yu, A. W., Song, D. and Zhou, D. (2020), 'Compositional Generalization via Neural-Symbolic Stack Machines'. arXiv:2008.06662 [cs]. http://arxiv.org/abs/2008.06662 (accessed on: 2026-01-31) (page 68).

Chen, Y., Wang, S., Hilgetag, C. C. and Zhou, C. (2013), 'Trade-off between Multiple Constraints Enables Simultaneous Formation of Modules and Hubs in Neural Systems', *PLOS Computational Biology* **9**(3), e1002937. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1002937 (page 45).

Cherniak, C. (2012), Chapter 17 - Neural wiring optimization, *in* M. A. Hofman and D. Falk, eds, 'Progress in Brain Research', Vol. 195 of *Evolution of the Primate Brain*, Elsevier, pp. 361–371. doi: 10.1016/B978-0-444-53860-4.00017-9 (page 45).

Chklovskii, D. B. (2004), 'Synaptic connectivity and neuronal morphology: two sides of the same coin', *Neuron* **43**(5), 609–617. doi: 10.1016/j.neuron.2004.08.012 (page 45).

Chklovskii, D. B., Schikorski, T. and Stevens, C. F. (2002), 'Wiring Optimization in Cortical Circuits', *Neuron* **34**(3), 341–347. doi: 10.1016/S0896-6273(02)00679-7 (page 45).

Chomsky, N. (1965), *Aspects of the theory of syntax*, Aspects of the theory of syntax, M.I.T. Press, Oxford, England. (page 66).

Chomsky, N. (1980), 'The new organology', *Behavioral and Brain Sciences* **3**(1), 42–61. doi: 10.1017/S0140525X0000176X (page 25).

Christensen, D. V. and others (2021), '2022 roadmap on neuromorphic computing and engineering', *Neuromorphic Computing and Engineering* **2**. (page 97).

Clune, J., Mouret, J.-B. and Lipson, H. (2013), 'The evolutionary origins of modularity', *Proceedings of the Royal Society B: Biological Sciences* **280**(1755), 20122863. Publisher: Royal Society. doi: 10.1098/rspb.2012.2863 (pages 35, 47, 69).

Cohen, G., Afshar, S., Tapson, J. and van Schaik, A. (2017), 'EMNIST: an extension of MNIST to handwritten letters'. arXiv:1702.05373 [cs]. http://arxiv.org/abs/1702.05373 (accessed on: 2023-07-25) (page 48).

Cook, M. (2004), 'Universality in elementary cellular automata', *Complex Systems* **15**(1), 1–40. (pages 95, 96, 115).

Csordás, R., Steenkiste, S. v. and Schmidhuber, J. (2021), 'Are Neural Nets Modular? Inspecting Functional Modularity Through Differentiable Weight Masks'. arXiv:2010.02066 [cs]. http://arxiv.org/abs/2010.02066 (accessed on: 2025-12-09) (pages 34, 47, 60).

Dalgaty, T., Moro, F., Demirağ, Y., De Pra, A., Indiveri, G., Vianello, E. and Payvand, M. (2024), 'Mosaic: in-memory computing and routing for small-world spike-based neuromorphic systems', *Nature Communications* **15**(1), 142. Publisher: Nature Publishing Group. doi: 10.1038/s41467-023-44365-x (pages 60, 77).

Damoiseaux, J. S. and Greicius, M. D. (2009), 'Greater than the sum of its parts: a review of studies combining structural connectivity and resting-state functional connectivity', *Brain Structure and Function* **213**(6), 525–533. doi: 10.1007/s00429-009-0208-6 (page 26).

Dehaene, S. (2020), *How we learn: why brains learn better than any machine ... for now*, Viking, New York, New York. (pages 29, 34).

Dehaene, S. and Cohen, L. (2007), 'Cultural recycling of cortical maps', *Neuron* **56**(2), 384–398. doi: 10.1016/j.neuron.2007.10.004 (page 29).

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A., Caron, M., Geirhos, R., Alabdulmohsin, I., Jenatton, R., Beyer, L., Tschannen, M., Arnab, A., Wang, X., Riquelme, C., Minderer, M., Puigcerver, J., Evci, U., Kumar, M., van Steenkiste, S., Elsayed, G. F., Mahendran, A., Yu, F., Oliver, A., Huot, F., Bastings, J., Collier, M. P., Gritsenko, A., Birodkar, V., Vasconcelos, C., Tay, Y., Mensink, T., Kolesnikov, A., Pavetić, F., Tran, D., Kipf, T., Lučić, M., Zhai, X., Keysers, D., Harmsen, J. and Houlsby, N. (2023), 'Scaling vision transformers to 22 billion parameters'. https://arxiv.org/abs/2302.05442 (page 121).

Deng, L. (2012), 'The mnist database of handwritten digit images for machine learning research', *IEEE Signal Processing Magazine* **29**(6), 141–142. (page 48).

Dobs, K., Martinez, J., Kell, A. J. E. and Kanwisher, N. (2022), 'Brain-like functional specialization emerges spontaneously in deep neural networks', *Science Advances* **8**(11), eabl8913. Publisher: American Association for the Advancement of Science. doi: 10.1126/sciadv.abl8913 (page 61).

Driscoll, L. N., Shenoy, K. and Sussillo, D. (2024), 'Flexible multitask computation in recurrent networks utilizes shared dynamical motifs', *Nature Neuroscience* **27**(7), 1349–1363. Publisher: Nature Publishing Group. doi: 10.1038/s41593-024-01668-6 (pages 34, 35).

Dwivedi, V. P. and Bresson, X. (2021), 'A generalization of transformer networks to graphs'. https://arxiv.org/abs/2012.09699 (pages 117, 131).

Ellefsen, K. O., Mouret, J.-B. and Clune, J. (2015), 'Neural Modularity Helps Organisms Evolve to Learn New Skills without Forgetting Old Skills', *PLOS Computational Biology* **11**(4), e1004128. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1004128 (pages 35, 61, 69).

Fakhar, K., Dixit, S., Hadaeghi, F., Kording, K. P. and Hilgetag, C. C. (2023), 'When Neural Activity Fails to Reveal Causal Contributions'. Pages: 2023.06.06.543895 Section: New Results. https://www.biorxiv.org/content/10.1101/2023.06.06.543895v1 (accessed on: 2023-07-10) (page 58).

Fakhar, K. and Hilgetag, C. C. (2022), 'Systematic perturbation of an artificial neural network: A step towards quantifying causal contributions in the brain', *PLOS Computational Biology* **18**(6), e1010250. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1010250 (pages 28, 46).

Faldor, M. and Cully, A. (2024), Toward artificial open-ended evolution within lenia using quality-diversity, *in* 'ALIFE 2024: Proceedings of the 2024 Artificial Life Conference', Artificial Life Conference Proceedings, p. 85. doi: 10.1162/isal_a_00827 (page 97).

Faldor, M. and Cully, A. (2025), CAX: Cellular automata accelerated in JAX, *in* 'The Thirteenth International Conference on Learning Representations (ICLR)'. Retrieved from https://openreview.net/forum?id=o2Igqm95SJ (pages 95, 97, 98).

Ferrier, D. (1875), 'Experiments on the brain of monkeys.—No. I', *Proceedings of the Royal Society of London* **23**(156-163), 409–430. doi: 10.1098/rspl.1874.0058 (page 24).

Filan, D., Casper, S., Hod, S., Wild, C., Critch, A. and Russell, S. (2021), 'Clusterability in Neural Networks'. arXiv:2103.03386 [cs]. http://arxiv.org/abs/2103.03386 (accessed on: 2024-05-17) (page 61).

Flake, G. W. (1998), *The computational beauty of nature*, MIT Press, Cambridge, MA, USA. (page 94).

Fodor, J. A. (1983), *The Modularity of Mind*, The MIT Press. doi: 10.7551/mitpress/4737.001.0001 (pages 25, 45).

Fodor, J. A. (2000), *The mind doesn't work that way: The scope and limits of computational psychology*, The mind doesn't work that way: The scope and limits of computational psychology, The MIT Press, Cambridge, MA, US. Pages: xi, 126. (page 25).

Fodor, J. A. and Pylyshyn, Z. W. (1988), 'Connectionism and cognitive architecture: A critical analysis', *Cognition* **28**(1), 3–71. doi: 10.1016/0010-0277(88)90031-5 (page 67).

Frankenhuis, W. E. and Ploeger, A. (2007), 'Evolutionary Psychology Versus Fodor: Arguments For and Against the Massive Modularity Hypothesis', *Philosophical Psychology* **20**(6), 687–710. Publisher: Routledge _eprint: https://doi.org/10.1080/09515080701665904. doi: 10.1080/09515080701665904 (page 26).

Frankle, J. and Carbin, M. (2019), 'The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks'. arXiv:1803.03635 [cs]. http://arxiv.org/abs/1803.03635 (accessed on: 2025-12-09) (page 33).

Fritsch, G. and Hitzig, E. (1870), 'Electric excitability of the cerebrum (Über die elektrische Erregbarkeit des Grosshirns)', *Epilepsy & Behavior* **15**(2), 123–130. Publisher: Elsevier. doi: 10.1016/j.yebeh.2009.03.001 (page 23).

Gajardo, A., Moreira, A. and Goles, E. (2002), 'Complexity of langton's ant', *Discrete Applied Mathematics* **117**(1), 41–50. doi: 10.1016/S0166-218X(00)00334-6 (page 96).

Gardner, M. (1970), 'Mathematical Games', *Scientific American* **223**(4), 120–123. Retrieved from http://www.jstor.org/stable/24927642 (page 96).

Geman, S., Bienenstock, E. and Doursat, R. (1992), 'Neural Networks and the Bias/Variance Dilemma', *Neural Computation* **4**(1), 1–58. doi: 10.1162/neco.1992.4.1.1 (page 66).

Gerstner, W., Kistler, W. M., Naud, R. and Paninski, L. (2014), *Neuronal dynamics: from single neurons to networks and models of cognition*, Cambridge University Press, Cambridge, United Kingdom. (page 47).

Gokhale, M., Graham, P., Johnson, E., Rollins, N. and Wirthlin, M. (2004), Dynamic reconfiguration for management of radiation-induced faults in FPGAs, *in* '18th International Parallel and Distributed Processing Symposium, 2004. Proceedings.', pp. 145–. doi: 10.1109/IPDPS.2004.1303127 (page 115).

Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y. and Schölkopf, B. (2020), 'RECURRENT INDEPENDENT MECHANISMS', p. 35. (page 61).

Grattarola, D., Livi, L. and Alippi, C. (2021), 'Learning graph cellular automata'. arXiv:2110.14237 [cs]. http://arxiv.org/abs/2110.14237 (accessed on: 2025-09-08) (page 116).

Guichard, E., Reimers, F., Kvalsund, M., Lepperød, M. and Nichele, S. (2025), 'Engramnca: a neural cellular automaton model of memory transfer', arXiv preprint arXiv:2504.11855. (page 97).

Ha, D. and Tang, Y. (2022), 'Collective Intelligence for Deep Learning: A Survey of Recent Developments'. arXiv:2111.14377 [cs]. http://arxiv.org/abs/2111.14377 (accessed on: 2025-12-11) (pages 32, 47).

Hamon, G., Etcheverry, M., Chan, B. W.-C., Moulin-Frier, C. and Oudeyer, P.-Y. (2024), 'Discovering sensorimotor agency in cellular automata using diversity search', arXiv preprint arXiv:2402.10236. (page 97).

Hayden, B. Y. (2022), 'The pernicious danger of cortical brain maps'. arXiv:2209.06740 [q-bio]. http://arxiv.org/abs/2209.06740 (accessed on: 2025-12-08) (page 29).

Hayden, B. Y., Heilbronner, S. R. and Yoo, S. B. M. (2025), 'Rethinking the centrality of brain areas in understanding functional organization', *Nature Neuroscience* pp. 1–12. Publisher: Nature Publishing Group. doi: 10.1038/s41593-025-02166-z (page 29).

He, J., Sun, J. and Deem, M. W. (2009), 'Spontaneous emergence of modularity in a model of evolving individuals and in real networks', *Physical Review E* **79**(3), 031907. Publisher: American Physical Society. doi: 10.1103/PhysRevE.79.031907 (page 35).

Heuvel, M. P. v. d. and Sporns, O. (2011), 'Rich-Club Organization of the Human Connectome', *Journal of Neuroscience* **31**(44), 15775–15786. Publisher: Society for Neuroscience Section: Articles. doi: 10.1523/JNEUROSCI.3539-11.2011 (page 45).

Hiesinger, P. R. (2021), *The self-assembling brain: how neural networks grow smarter*, Princeton university press, Princeton (N.J.). (page 37).

Hod, S., Filan, D., Casper, S., Critch, A. and Russell, S. (2022), 'Quantifying Local Specialization in Deep Neural Networks'. arXiv:2110.08058 [cs]. http://arxiv.org/abs/2110.08058 (accessed on: 2025-12-10) (pages 34, 61).

Horvát, S., Gămănuț, R., Ercsey-Ravasz, M., Magrou, L., Gămănuț, B., Essen, D. C. V., Burkhalter, A., Knoblauch, K., Toroczkai, Z. and Kennedy, H. (2016), 'Spatial Embedding and Wiring Cost Constrain the Functional Layout of the Cortical Network of Rodents and Primates', *PLOS Biology* **14**(7), e1002512. Publisher: Public Library of Science. doi: 10.1371/journal.pbio.1002512 (page 28).

Huizinga, J., Mouret, J.-B. and Clune, J. (2014), Evolving Neural Networks That Are Both Modular and Regular: HyperNeat Plus the Connection Cost Technique. doi: 10.1145/2576768.2598232 (page 36).

Huizinga, J., Mouret, J.-B. and Clune, J. (2016), Does Aligning Phenotypic and Genotypic Modularity Improve the Evolution of Neural Networks?, *in* 'Proceedings of the Genetic and Evolutionary Computation Conference 2016', GECCO '16, Association for Computing Machinery, pp. 125–132. doi: 10.1145/2908812.2908836 (page 36).

Hupkes, D., Dankers, V., Mul, M. and Bruni, E. (2020), 'Compositionality Decomposed: How do Neural Networks Generalise?', *Journal of Artificial Intelligence Research* **67**, 757–795. doi: 10.1613/jair.1.11674 (page 67).

Jacobs, R. and Jordan, M. (1992), 'Computational Consequences of a Bias toward Short Connections', *Journal of cognitive neuroscience* **4**, 323–36. doi: 10.1162/jocn.1992.4.4.323 (pages 35, 69).

Jaegle, A., Borgeaud, S., Alayrac, J.-B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., Hénaff, O., Botvinick, M. M., Zisserman, A., Vinyals, O. and Carreira, J. (2022), 'Perceiver io: A general architecture for structured inputs & outputs'. https://arxiv.org/abs/2107.14795 (pages 122, 131).

Johnston, W. J. and Fusi, S. (2023), 'Abstract representations emerge naturally in neural networks trained to perform multiple tasks', *Nature Communications* **14**(1), 1040. Publisher: Nature Publishing Group. doi: 10.1038/s41467-023-36583-0 (page 34).

Johnston, W. J. and Fusi, S. (2024), 'Modular representations emerge in neural networks trained to perform context-dependent tasks'. Pages: 2024.09.30.615925 Section: New Results. https://www.biorxiv.org/content/10.1101/2024.09.30.615925v2 (accessed on: 2024-10-14) (page 35).

Jonas, E. and Kording, K. P. (2016), 'Could a Neuroscientist Understand a Microprocessor?'. Pages: 055624 Section: New Results. https://www.biorxiv.org/content/10.1101/055624v2 (accessed on: 2023-04-18) (page 46).

Jonas, E. and Kording, K. P. (2017), 'Could a Neuroscientist Understand a Microprocessor?', *PLOS Computational Biology* **13**(1), e1005268. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1005268 (page 28).

Kaiser, M. and Hilgetag, C. C. (2004), 'Spatial Growth of Real-world Networks', *Physical Review E* **69**(3), 036103. arXiv:cond-mat/0312037. doi: 10.1103/PhysRevE.69.036103 (page 28).

Kaiser, M. and Hilgetag, C. C. (2006), 'Nonoptimal Component Placement, but Short Processing Paths, due to Long-Distance Projections in Neural Systems', *PLOS Computational Biology* **2**(7), e95. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.0020095 (pages 28, 45).

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J. and Amodei, D. (2020), 'Scaling Laws for Neural Language Models'. arXiv:2001.08361 [cs]. http://arxiv.org/abs/2001.08361 (accessed on: 2026-01-27) (page 68).

Kashtan, N., Noor, E. and Alon, U. (2007), 'Varying environments can speed up evolution', *Proceedings of the National Academy of Sciences* **104**(34), 13711–13716. Publisher: National Academy of Sciences Section: Biological Sciences. doi: 10.1073/pnas.0611630104 (pages 35, 61).

Ke, N. R., Didolkar, A., Mittal, S., Goyal, A., Lajoie, G., Bauer, S., Rezende, D., Bengio, Y., Mozer, M. and Pal, C. (2021), 'Systematic Evaluation of Causal Discovery in Visual Model Based Reinforcement Learning'. arXiv:2107.00848 [cs, stat]. http://arxiv.org/abs/2107.00848 (accessed on: 2023-03-17) (page 46).

Keirsbilck, M. V., Keller, A. and Yang, X. (2019), 'Rethinking Full Connectivity in Recurrent Neural Networks'. arXiv:1905.12340 [cs]. http://arxiv.org/abs/1905.12340 (accessed on: 2025-12-05) (page 36).

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., Zee, M. v. and Bousquet, O. (2020), 'Measuring Compositional Generalization: A Comprehensive Method on Realistic Data'. arXiv:1912.09713 [cs]. http://arxiv.org/abs/1912.09713 (accessed on: 2025-05-05) (page 67).

Khona, M., Chandra, S., Ma, J. J. and Fiete, I. R. (2023), 'Winning the Lottery With Neural Connectivity Constraints: Faster Learning Across Cognitive Tasks With Spatially Constrained Sparse RNNs', *Neural Computation* **35**(11), 1850–1869. doi: 10.1162/neco_a_01613 (pages 36, 69, 77).

Kim, N. and Linzen, T. (2020), COGS: A Compositional Generalization Challenge Based on Semantic Interpretation, *in* B. Webber, T. Cohn, Y. He and Y. Liu, eds, 'Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)', Association for Computational Linguistics, pp. 9087–9105. doi: 10.18653/v1/2020.emnlp-main.731 (page 67).

Kingma, D. P. and Ba, J. (2014), 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980* . (page 124).

Kirsch, L., Harrison, J., Sohl-Dickstein, J. and Metz, L. (2024), 'General-Purpose In-Context Learning by Meta-Learning Transformers'. arXiv:2212.04458 [cs]. http://arxiv.org/abs/2212.04458 (accessed on: 2025-06-18) (page 117).

Kirsch, L., Kunze, J. and Barber, D. (2018), 'Modular Networks: Learning to Decompose Neural Computation', *arXiv:1811.05249 [cs, stat]* . arXiv: 1811.05249. Retrieved from http://arxiv.org/abs/1811.05249 (page 61).

Kirsch, L. and Schmidhuber, J. (2022), 'Meta Learning Backpropagation And Improving It'. arXiv:2012.14905 [cs, stat]. http://arxiv.org/abs/2012.14905 (accessed on: 2024-08-23) (page 117).

Kirschner, M. W., Gerhart, J. C. and Norton, J. (2005), *The Plausibility of Life: Resolving Darwin's Dilemma*, Yale University Press. Retrieved from https://www.jstor.org/stable/j.ctt1np9wb (page 19).

Kolmogorov, A. N. (1968), 'Three approaches to the quantitative definition of information *', *International Journal of Computer Mathematics* **2**(1-4), 157–168. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00207166808803030. doi: 10.1080/00207166808803030 (page 37).

Konrad-Lorenz-Institute (2005), *Modularity: Understanding the Development and Evolution of Natural Complex Systems*, The MIT Press. doi: 10.7551/mitpress/4734.001.0001 (pages 19, 20, 22).

Kornblith, S., Norouzi, M., Lee, H. and Hinton, G. (2019), 'Similarity of Neural Network Representations Revisited'. arXiv:1905.00414 [cs, q-bio, stat]. http://arxiv.org/abs/1905.00414 (accessed on: 2024-02-26) (page 52).

Koulakov, A., Shuvaev, S., Lachi, D. and Zador, A. (2022), 'Encoding innate ability through a genomic bottleneck'. Pages: 2021.03.16.435261 Section: New Results. https://www.biorxiv.org/content/10.1101/2021.03.16.435261v2 (accessed on: 2023-04-12) (page 37).

Kuo, Y.-L., Katz, B. and Barbu, A. (2021), 'Compositional Networks Enable Systematic Generalization for Grounded Language Understanding'. arXiv:2008.02742 [cs]. http://arxiv.org/abs/2008.02742 (accessed on: 2026-02-02) (page 68).

Lake, B. M. and Baroni, M. (2018), 'Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks'. arXiv:1711.00350. http://arxiv.org/abs/1711.00350 (accessed on: 2024-10-28) (page 67).

Lake, B. M., Ullman, T. D., Tenenbaum, J. B. and Gershman, S. J. (2016), 'Building Machines That Learn and Think Like People'. arXiv:1604.00289 [cs]. http://arxiv.org/abs/1604.00289 (accessed on: 2026-01-25) (page 67).

Lashley, K. S. (1930), 'Basic neural mechanisms in behavior', *Psychological Review* **37**(1), 1–24. Place: US Publisher: Psychological Review Company. doi: 10.1037/h0074134 (page 24).

Lepori, M. A., Serre, T. and Pavlick, E. (2023), Break It Down: Evidence for Structural Compositionality in Neural Networks. Retrieved from https://openreview.net/forum?id=rwbzMiuFQl (pages 34, 67).

Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J. and Hinton, G. (2020), 'Backpropagation and the brain', *Nature Reviews Neuroscience* **21**(6), 335–346. Number: 6 Publisher: Nature Publishing Group. doi: 10.1038/s41583-020-0277-3 (page 47).

Lipson, H. (2007), 'Principles of modularity, regularity, and hierarchy for scalable systems', *Journal of Biological Physics and Chemistry* **7**(4), 125–128. doi: 10.4024/40701.jbpc.07.04 (page 20).

Liu, Z., Gan, E. and Tegmark, M. (2023), 'Seeing is Believing: Brain-Inspired Modular Training for Mechanistic Interpretability'. arXiv:2305.08746 [cs]. http://arxiv.org/abs/2305.08746 (accessed on: 2025-11-29) (pages 36, 60).

Malach, E., Yehudai, G., Shalev-Shwartz, S. and Shamir, O. (2020), 'Proving the Lottery Ticket Hypothesis: Pruning is All You Need'. arXiv:2002.00585 [cs, stat]. http://arxiv.org/abs/2002.00585 (accessed on: 2022-12-05) (page 33).

Marr, D. (1976), 'Early processing of visual information', *Philosophical Transactions of the Royal Society of London. B, Biological Sciences* **275**(942), 483–519. Publisher: Royal Society. doi: 10.1098/rstb.1976.0090 (page 20).

Marton, C. D., Lajoie, G. and Rajan, K. (2021), 'Efficient and robust multi-task learning in the brain with modular task primitives', *arXiv:2105.14108 [cs, q-bio]* . arXiv: 2105.14108. Retrieved from http://arxiv.org/abs/2105.14108 (page 47).

Mayr, E. (1961), 'Cause and Effect in Biology', *Science* **134**(3489), 1501–1506. Publisher: American Association for the Advancement of Science. Retrieved from https://www.jstor.org/stable/1707986 (page 21).

Mead, C. (1990), 'Neuromorphic electronic systems', *Proceedings of the IEEE* **78**(10), 1629–1636. doi: 10.1109/5.58356 (page 97).

Mehta, R. (2019), 'Sparse Transfer Learning via Winning Lottery Tickets'. arXiv:1905.07785 [cs]. http://arxiv.org/abs/1905.07785 (accessed on: 2025-12-10) (page 33).

Mengistu, H., Huizinga, J., Mouret, J.-B. and Clune, J. (2016), 'The Evolutionary Origins of Hierarchy', *PLOS Computational Biology* **12**(6), e1004829. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1004829 (pages 35, 69).

Meunier, D., Lambiotte, R. and Bullmore, E. (2010), 'Modular and hierarchically modular organization of brain networks', *Frontiers in Neuroscience* **4**, 200. doi: 10.3389/fnins.2010.00200 (page 97).

Meunier, D., Lambiotte, R., Fornito, A., Ersche, K. D. and Bullmore, E. T. (2009), 'Hierarchical Modularity in Human Brain Functional Networks', *Frontiers in Neuroinformatics* **3**, 37. doi: 10.3389/neuro.11.037.2009 (page 26).

Meyes, R., Puiseau, C. W. d., Posada-Moreno, A. and Meisen, T. (2020), 'Under the Hood of Neural Networks: Characterizing Learned Representations by Functional Neuron Populations and Network Ablations'. arXiv:2004.01254 [cs]. http://arxiv.org/abs/2004.01254 (accessed on: 2025-12-10) (page 46).

Miotti, Pietro and Niklasson, Eyvind and Randazzo, Ettore and Mordvintsev, Alexander (2025), 'Differentiable logic CA: from game of life to pattern generation', https://google-research.github.io/self-organising-systems/difflogic-ca. (page 97).

Mitchell, M., Crutchfield, J. and Das, R. (2000), 'Evolving Cellular Automata with Genetic Algorithms: A Review of Recent Work', *First Int. Conf. on Evolutionary Computation and Its Applications* **1**. (page 96).

Mittal, S., Bengio, Y. and Lajoie, G. (2022), 'Is a Modular Architecture Enough?', *arXiv:2206.02713 [cs]* . arXiv: 2206.02713. Retrieved from http://arxiv.org/abs/2206.02713 (page 60).

Morcos, A. S., Yu, H., Paganini, M. and Tian, Y. (2019), 'One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers'. arXiv:1906.02773 [stat]. http://arxiv.org/abs/1906.02773 (accessed on: 2025-12-10) (page 33).

Mordvintsev, A., Randazzo, E., Niklasson, E. and Levin, M. (2020), 'Growing Neural Cellular Automata', *Distill* **5**(2), e23. doi: 10.23915/distill.00023 (pages 37, 95, 97, 115, 125).

Musall, S., Kaufman, M. T., Juavinett, A. L., Gluf, S. and Churchland, A. K. (2019), 'Single-trial neural dynamics are dominated by richly varied movements', *Nature Neuroscience* **22**(10), 1677–1686. Publisher: Nature Publishing Group. doi: 10.1038/s41593-019-0502-4 (page 29).

Musall, S., Urai, A. E., Sussillo, D. and Churchland, A. K. (2019), 'Harnessing behavioral diversity to understand neural computations for cognition', *Current Opinion in Neurobiology* **58**, 229–238. doi: 10.1016/j.conb.2019.09.011 (page 47).

Mészáros, B., Knight, J. C., Akarca, D. and Nowotny, T. (2025), 'Space as Time Through Neuron Position Learning'. arXiv:2511.01632 [cs]. http://arxiv.org/abs/2511.01632 (accessed on: 2025-12-05) (page 36).

Najarro, E., Sudhakaran, S., Glanois, C. and Risi, S. (2022*a*), 'HyperNCA: Growing Developmental Networks with Neural Cellular Automata'. arXiv:2204.11674 [cs]. http://arxiv.org/abs/2204.11674 (accessed on: 2025-12-05) (page 38).

Najarro, E., Sudhakaran, S., Glanois, C. and Risi, S. (2022*b*), 'HyperNCA: Growing developmental networks with neural cellular automata', arXiv preprint arXiv:2204.11674. (pages 95, 97).

Najarro, E., Sudhakaran, S. and Risi, S. (2023), 'Towards Self-Assembling Artificial Neural Networks through Neural Developmental Programs'. arXiv:2307.08197 [cs]. http://arxiv.org/abs/2307.08197 (accessed on: 2025-12-05) (page 38).

Newman, M. E. J. (2006), 'Modularity and community structure in networks', *Proceedings of the National Academy of Sciences* **103**(23), 8577–8582. Publisher: National Academy of Sciences Section: Physical Sciences. doi: 10.1073/pnas.0601602103 (pages 19, 30, 45, 51, 58, 85, 139).

Noble, S., Curtiss, J., Pessoa, L. and Scheinost, D. (2024), 'The tip of the iceberg: A call to embrace anti-localizationism in human neuroscience research', *Imaging Neuroscience* **2**, imag–2–00138. doi: 10.1162/imag_a_00138 (page 29).

Nudo, R. J. (2013), 'Recovery after brain injury: mechanisms and principles', *Frontiers in Human Neuroscience* **7**, 887. doi: 10.3389/fnhum.2013.00887 (page 114).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830. (page 82).

Pessoa, L. (2022), *The entangled brain: how perception, cognition, and emotion are woven together*, The MIT Press, Cambridge, Massachusetts. (pages 28, 44, 61).

Pessoa, L., Medina, L. and Desfilis, E. (2021), 'Refocusing neuroscience: moving away from mental categories and towards complex behaviours', *Philosophical Transactions of the Royal Society B: Biological Sciences* **377**(1844), 20200534. doi: 10.1098/rstb.2020.0534 (page 29).

Petersen, F., Borgelt, C., Kuehne, H. and Deussen, O. (2022), 'Deep Differentiable Logic Gate Networks'. arXiv:2210.08277 [cs]. `http://arxiv.org/abs/2210.08277` (accessed on: 2025-04-29) (pages 116, 117).

Pfeiffer, J., Ruder, S., Vulić, I. and Ponti, E. M. (2023), 'Modular Deep Learning'. arXiv:2302.11529 [cs]. `http://arxiv.org/abs/2302.11529` (accessed on: 2023-05-09) (page 47).

Pfeiffer, J., Ruder, S., Vulić, I. and Ponti, E. M. (2024), 'Modular Deep Learning'. arXiv:2302.11529 [cs]. `http://arxiv.org/abs/2302.11529` (accessed on: 2025-12-11) (page 32).

Power, J. D., Cohen, A. L., Nelson, S. M., Wig, G. S., Barnes, K. A., Church, J. A., Vogel, A. C., Laumann, T. O., Miezin, F. M., Schlaggar, B. L. and Petersen, S. E. (2011), 'Functional Network Organization of the Human Brain', *Neuron* **72**(4), 665–678. doi: 10.1016/j.neuron.2011.09.006 (page 46).

Preti, M. G. and Van De Ville, D. (2019), 'Decoupling of brain function from structure reveals regional behavioral specialization in humans', *Nature Communications* **10**(1), 4747. Publisher: Nature Publishing Group. doi: 10.1038/s41467-019-12765-7 (page 27).

Rafler, S. (2011), 'Generalization of conway's "game of life" to a continuous domain - SmoothLife', arXiv preprint arXiv:1111.1567. (pages 95, 97, 115).

Raj, A. and Chen, Y.-h. (2011), 'The Wiring Economy Principle: Connectivity Determines Anatomy in the Human Brain', *PLOS ONE* **6**(9), e14832. Publisher: Public Library of Science. doi: 10.1371/journal.pone.0014832 (page 45).

Ramanujan, V., Wortsman, M., Kembhavi, A., Farhadi, A. and Rastegari, M. (2020), 'What's Hidden in a Randomly Weighted Neural Network?'. arXiv:1911.13299 [cs]. http://arxiv.org/abs/1911.13299 (accessed on: 2025-12-09) (page 33).

Randazzo, E., Mordvintsev, A., Niklasson, E., Levin, M. and Greydanus, S. (2020), 'Self-classifying mnist digits', *Distill* . https://distill.pub/2020/selforg/mnist. doi: 10.23915/distill.00027.002 (pages 95, 97).

Redhardt, F., Akram, Y. and Schug, S. (2025), 'Scaling can lead to compositional generalization'. arXiv:2507.07207 [cs]. http://arxiv.org/abs/2507.07207 (accessed on: 2025-12-15) (page 68).

Reinke, C., Etcheverry, M. and Oudeyer, P.-Y. (2020), 'Intrinsically motivated discovery of diverse patterns in self-organizing systems', arXiv preprint arXiv:1908.06663. (page 97).

Rendell, P. (2016), *Turing Machine Universality of the Game of Life*, Vol. 18 of *Emergence, Complexity and Computation*, Springer International Publishing. doi: 10.1007/978-3-319-19842-2 (pages 95, 96).

Roberts, J. A., Perry, A., Lord, A. R., Roberts, G., Mitchell, P. B., Smith, R. E., Calamante, F. and Breakspear, M. (2016), 'The contribution of geometry to the human connectome', *NeuroImage* **124**, 379–393. doi: 10.1016/j.neuroimage.2015.09.009 (page 28).

Rosen, M. C. and Freedman, D. J. (2025), 'How distributed is the brain-wide network that is recruited for cognition?', *Nature Reviews Neuroscience* pp. 1–13. Publisher: Nature Publishing Group. doi: 10.1038/s41583-025-00992-5 (page 29).

Rosenbaum, C., Cases, I., Riemer, M. and Klinger, T. (2019), 'Routing Networks and the Challenges of Modular and Compositional Computation'. arXiv:1904.12774 [cs]. http://arxiv.org/abs/1904.12774 (accessed on: 2025-12-09) (page 32).

Rosenbaum, C., Klinger, T. and Riemer, M. (2017), 'Routing Networks: Adaptive Selection of Non-linear Functions for Multi-Task Learning'. arXiv:1711.01239 [cs]. http://arxiv.org/abs/1711.01239 (accessed on: 2025-12-09) (page 32).

Ruis, L., Andreas, J., Baroni, M., Bouchacourt, D. and Lake, B. M. (2020), 'A Benchmark for Systematic Generalization in Grounded Language Understanding'. arXiv:2003.05161 [cs]. http://arxiv.org/abs/2003.05161 (accessed on: 2025-05-05) (page 68).

Salatiello, A. (2025), Modularity is the Bedrock of Natural and Artificial Intelligence. Retrieved from https://openreview.net/forum?id=kqrnhp3nNS (page 32).

Samu, D., Seth, A. K. and Nowotny, T. (2014), 'Influence of Wiring Cost on the Large-Scale Architecture of Human Cortical Connectivity', *PLOS Computational Biology* **10**(4), e1003557. Publisher: Public Library of Science. doi: 10.1371/journal.pcbi.1003557 (page 28).

Sandbrink, K., Bauer, J. P., Proca, A. M., Saxe, A. M., Summerfield, C. and Hummos, A. (2024), 'Flexible task abstractions emerge in linear networks with fast and bounded units'. arXiv:2411.03840. http://arxiv.org/abs/2411.03840 (accessed on: 2024-12-02) (page 34).

Sapin, E., Bailleux, O. and Chabrier, J.-J. (2003), Research of a Cellular Automaton Simulating Logic Gates by Evolutionary Algorithms, *in* 'Evolvable Systems: From Biology to Hardware'. doi: 10.1007/3-540-36599-0_39 (page 96).

Sauro, H. M. (2008), 'Modularity defined', *Molecular Systems Biology* **4**(1), 166. Publisher: John Wiley & Sons, Ltd. doi: 10.1038/msb.2008.3 (page 19).

Schuman, C. D., Potok, T. E., Patton, R. M., Birdwell, J. D., Dean, M. E., Rose, G. S. and Plank, J. S. (2017), 'A survey of neuromorphic computing and neural networks in hardware', arXiv preprint arXiv:1705.06963. (page 97).

Shallice, T. and Cooper, R. P. (2011), *The organisation of mind*, Oxford University Press, Oxford ; New York. OCLC: ocn664323812. (page 45).

Sheeran, C., Ham, A. S., Astle, D. E., Achterberg, J. and Akarca, D. (2024), 'Spatial embedding promotes a specific form of modularity with low entropy and heterogeneous spectral dynamics'. arXiv:2409.17693 [cs]. http://arxiv.org/abs/2409.17693 (accessed on: 2025-12-05) (page 77).

Shine, J. M., Bissett, P. G., Bell, P. T., Koyejo, O., Balsters, J. H., Gorgolewski, K. J., Moodie, C. A. and Poldrack, R. A. (2016), 'The Dynamics of Functional Brain Networks: Integrated Network States during Cognitive Task Performance', *Neuron* **92**(2), 544–554. Publisher: Elsevier. doi: 10.1016/j.neuron.2016.09.018 (page 58).

Simon, H. A. (1962), 'The Architecture of Complexity', *Proceedings of the American Philosophical Society* **106**(6), 467–482. Publisher: American Philosophical Society. Retrieved from https://www.jstor.org/stable/985254 (page 19).

Simon, H. A. (1977), *Models of discovery and other topics in the methods of science*, number v. 54 *in* 'Boston studies in the philosophy of science', D. Reidel Pub. Co, Dordrecht, Holland ; Boston. (page 19).

Simpson, G. G. (1953), 'THE BALDWIN EFFECT', *Evolution* **7**(2), 110–117. doi: 10.1111/j.1558-5646.1953.tb00069.x (page 30).

Sinha, S., Premsri, T. and Kordjamshidi, P. (2024), 'A Survey on Compositional Learning of AI Models: Theoretical and Experimental Practices'. arXiv:2406.08787 [cs]. http://arxiv.org/abs/2406.08787 (accessed on: 2025-04-22) (page 67).

Soelen, R. V. and Sheppard, J. W. (2019), Using Winning Lottery Tickets in Transfer Learning for Convolutional Neural Networks, *in* '2019 International Joint Conference on Neural Networks (IJCNN)', pp. 1–8. ISSN: 2161-4407. doi: 10.1109/IJCNN.2019.8852405 (page 33).

Solomonoff, R. J. (1964), 'A formal theory of inductive inference. Part I', *Information and Control* **7**(1), 1–22. doi: 10.1016/S0019-9958(64)90223-2 (page 37).

Solé, R. V., Ferrer-Cancho, R., Montoya, J. M. and Valverde, S. (2002), 'Selection, tinkering, and emergence in complex networks', *Complexity* **8**(1), 20–33. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cplx.10055. doi: 10.1002/cplx.10055 (page 19).

Sporns, O. (2011), 'The human connectome: a complex network', *Annals of the New York Academy of Sciences* **1224**(1), 109–125. _eprint: https://nyaspubs.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1749-6632.2010.05888.x. doi: 10.1111/j.1749-6632.2010.05888.x (page 26).

Sporns, O. (2013), 'Network attributes for segregation and integration in the human brain', *Current Opinion in Neurobiology* **23**(2), 162–171. doi: 10.1016/j.conb.2012.11.015 (page 46).

Sporns, O. (2018), 'Graph theory methods: applications in brain networks', *Dialogues in Clinical Neuroscience* **20**(2), 111–121. doi: 10.31887/DCNS.2018.20.2/osporns (page 26).

Sporns, O. and Betzel, R. F. (2016), 'Modular Brain Networks', *Annual review of psychology* **67**, 613–640. doi: 10.1146/annurev-psych-122414-033634 (pages 26, 46).

Stanley, K., D'Ambrosio, D. and Gauci, J. (2009), 'A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks', *Artificial Life* **15**(2), 185–212. doi: 10.1162/artl.2009.15.2.15202 (page 36).

Striedter, G. F. (2005), *Principles of brain evolution*, Principles of brain evolution, Sinauer Associates, Sunderland, MA, US. Pages: xii, 436. (page 45).

Stringer, C., Pachitariu, M., Steinmetz, N., Reddy, C. B., Carandini, M. and Harris, K. D. (2019), 'Spontaneous behaviors drive multidimensional, brainwide activity', *Science* **364**(6437), eaav7893. Publisher: American Association for the Advancement of Science. doi: 10.1126/science.aav7893 (page 29).

Stöckl, C., Lang, D. and Maass, W. (2022), 'Structure induces computational function in networks with diverse types of spiking neurons'. Pages: 2021.05.18.444689 Section: New Results. https://www.biorxiv.org/content/10.1101/2021.05.18.444689v4 (accessed on: 2023-04-12) (page 37).

Subramanian, S., Bogin, B., Gupta, N., Wolfson, T., Singh, S., Berant, J. and Gardner, M. (2020), 'Obtaining Faithful Interpretations from Compositional Neural Networks'. arXiv:2005.00724 [cs]. http://arxiv.org/abs/2005.00724 (accessed on: 2025-04-22) (page 68).

Suh, N. P. (1990), *The principles of design*, number 6 *in* 'Oxford series on advanced manufacturing', Oxford Univ. Press, New York. (page 20).

Sun, P., Achterberg, J., Su, Z., Goodman, D. F. M. and Akarca, D. (2025), 'Exploiting heterogeneous delays for efficient computation in low-bit neural networks'. arXiv:2510.27434 [cs]. http://arxiv.org/abs/2510.27434 (accessed on: 2025-12-19) (page 36).

Sunada, S., Niiyama, T., Kanno, K., Nogami, R., Röhm, A., Awano, T. and Uchida, A. (2025), 'Blending Optimal Control and Biologically Plausible Learning for Noise-Robust Physical Neural Networks', *Physical Review Letters* **134**(1), 017301. Publisher: American Physical Society. doi: 10.1103/PhysRevLett.134.017301 (page 117).

Suárez, L. E., Markello, R. D., Betzel, R. F. and Misic, B. (2020), 'Linking Structure and Function in Macroscale Brain Networks', *Trends in Cognitive Sciences* **24**(4), 302–315. doi: 10.1016/j.tics.2020.01.008 (pages 26, 27).

Tay, Y., Dehghani, M., Bahri, D. and Metzler, D. (2022), 'Efficient transformers: A survey'. https://arxiv.org/abs/2009.06732 (page 132).

Thomas Yeo, B. T., Krienen, F. M., Sepulcre, J., Sabuncu, M. R., Lashkari, D., Hollinshead, M., Roffman, J. L., Smoller, J. W., Zöllei, L., Polimeni, J. R., Fischl, B., Liu, H. and Buckner, R. L. (2011), 'The organization of the human cerebral cortex estimated by intrinsic functional connectivity', *Journal of Neurophysiology* **106**(3), 1125–1165. Publisher: American Physiological Society. doi: 10.1152/jn.00338.2011 (page 26).

Tishby, N., Pereira, F. C. and Bialek, W. (2000), 'The information bottleneck method'. arXiv:physics/0004057. http://arxiv.org/abs/physics/0004057 (accessed on: 2023-03-22) (page 52).

Turing, A. M. (1937), 'On computable numbers, with an application to the Entscheidungsproblem', *Proceedings of the London Mathematical Society* **42**(1), 230–265. (page 96).

Ulmann, B. (2022), *Analog Computing*, De Gruyter Oldenbourg, Berlin, Boston. doi: 10.1515/9783110787740 (page 97).

van den Heuvel, M. P., Kahn, R. S., Goñi, J. and Sporns, O. (2012), 'High-cost, high-capacity backbone for global brain communication', *Proceedings of the National Academy of Sciences* **109**(28), 11372–11377. Publisher: Proceedings of the National Academy of Sciences. doi: 10.1073/pnas.1203593109 (page 26).

van der Plas, T. L., Tubiana, J., Le Goc, G., Migault, G., Kunst, M., Baier, H., Bormuth, V., Englitz, B. and Debrégeas, G. (2023), 'Neural assemblies uncovered by generative modeling explain whole-brain activity statistics and reflect structural connectivity', *eLife* **12**, e83139. Publisher: eLife Sciences Publications, Ltd. doi: 10.7554/eLife.83139 (page 46).

Vapnik, V. N. and Chervonenkis, A. Y. (1971), 'On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities', *Theory of Probability & Its Applications* **16**(2), 264–280. Publisher: Society for Industrial and Applied Mathematics. doi: 10.1137/1116025 (page 66).

Vishwanathan, A., Ramirez, A., Wu, J., Sood, A., Yang, R., Kemnitz, N., Ih, D., Turner, N., Lee, K., Tartavull, I., Silversmith, W., Jordan, C., David, C., Bland, D., Goldman, M., Aksay, E. and Seung, H. (2020), *Modularity and neural coding from a brainstem synaptic wiring diagram*. doi: 10.1101/2020.10.28.359620 (page 46).

von Neumann, J. (1956), *Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components*, Princeton University Press, Princeton, pp. 43–98. doi: doi:10.1515/9781400882618-003 (page 114).

von Neumann, J. (1966), 'Theory of Self-Reproducing Automata', *University of Illinois Press* . (page 114).

Vértes, P. E., Alexander-Bloch, A. F., Gogtay, N., Giedd, J. N., Rapoport, J. L. and Bullmore, E. T. (2012), 'Simple models of human brain functional networks', *Proceedings of the National Academy of Sciences* **109**(15), 5868–5873. Company: National Academy of Sciences Distributor: National Academy of Sciences Institution: National Academy of Sciences Label: National Academy of Sciences Publisher: Proceedings of the National Academy of Sciences. doi: 10.1073/pnas.1111738109 (page 28).

Wagner, G., Mezey, J. and Calabretta, R. (2005), Natural Selection and the Origin of Modules, pp. 33–50. doi: 10.7551/mitpress/4734.003.0009 (page 20).

Wagner, G. P., Pavlicev, M. and Cheverud, J. M. (2007), 'The road to modularity', *Nature Reviews Genetics* **8**(12), 921–931. Bandiera_abtest: a Cg_type: Nature Research Journals Number: 12 Primary_atype: Reviews Publisher: Nature Publishing Group. doi: 10.1038/nrg2267 (page 20).

Watanabe, C. (2018), 'Interpreting Layered Neural Networks via Hierarchical Modular Representation'. arXiv:1810.01588 [cs, stat]. `http://arxiv.org/abs/1810.01588` (accessed on: 2024-03-03) (page 45).

Watts, D. J. and Strogatz, S. H. (1998), 'Collective dynamics of 'small-world' networks', *Nature* **393**(6684), 440–442. Publisher: Nature Publishing Group. doi: 10.1038/30918 (pages 19, 26).

Weisstein, E. W. (n.d.), 'Universal cellular automaton', MathWorld–A Wolfram Web Resource. Accessed: 2025-04-01. https://mathworld.wolfram.com/UniversalCellularAutomaton.html (page 96).

Wernicke, C. (1874), Der aphasische Symptomenkomplex, *in* C. Wernicke, ed., 'Der aphasische Symptomencomplex: Eine psychologische Studie auf anatomischer Basis', Springer, Berlin, Heidelberg, pp. 1–70. doi: 10.1007/978-3-642-65950-8_1 (page 23).

Whidden, P. (2020), 'Growing neural cellular automata pytorch implementation', https://github.com/PWhiddy/Growing-Neural-Cellular-Automata-Pytorch. (page 97).

Whittington, J. C. R. and Bogacz, R. (2019), 'Theories of Error Back-Propagation in the Brain', *Trends in Cognitive Sciences* **23**(3), 235–250. Publisher: Elsevier. doi: 10.1016/j.tics.2018.12.005 (page 47).

Whittington, J. C. R., Muller, T. H., Mark, S., Barry, C. and Behrens, T. E. J. (2018), 'Generalisation of structural knowledge in the hippocampal-entorhinal system'. arXiv:1805.09042 [cs, q-bio, stat]. http://arxiv.org/abs/1805.09042 (accessed on: 2023-07-03) (page 47).

Wolfram, S. (1984), 'Cellular automata as models of complexity', *Nature* **311**(5985), 419–424. doi: 10.1038/311419a0 (page 95).

Wolfram, S. (2002), *A New Kind of Science*, Wolfram Media. (pages 96, 115).

Woods, R. and Lightbody, G. (2008), Robustness in Digital Hardware, *in* A. Schuster, ed., 'Robust Intelligent Systems', Springer London, London, pp. 3–21. doi: 10.1007/978-1-84800-261-6_1 (pages 114, 115).

Wortsman, M., Ramanujan, V., Liu, R., Kembhavi, A., Rastegari, M., Yosinski, J. and Farhadi, A. (2020), 'Supermasks in Superposition', *arXiv:2006.14769 [cs, stat]* . arXiv: 2006.14769. Retrieved from http://arxiv.org/abs/2006.14769 (page 34).

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L. and Liu, T.-Y. (2020), 'On layer normalization in the transformer architecture'. https://arxiv.org/abs/2002.04745 (page 121).

Yamins, D. L. K. and DiCarlo, J. J. (2016), 'Using goal-driven deep learning models to understand sensory cortex', *Nature Neuroscience* **19**(3), 356–365. Number: 3 Publisher: Nature Publishing Group. doi: 10.1038/nn.4244 (page 47).

Yang, G. R. (2019), 'Task representations in neural networks trained to perform many cognitive tasks', *Nature Neuroscience* **22**, 16. (pages 34, 81).

Yang, R., Xu, H., Wu, Y. and Wang, X. (2020), 'Multi-Task Reinforcement Learning with Soft Modularization', *arXiv:2003.13661 [cs, stat]* . arXiv: 2003.13661. Retrieved from http://arxiv.org/abs/2003.13661 (page 61).

Young, M. P., Young, M. P., Hilgetag, C. and Scannell, J. W. (2000), 'On imputing function to structure from the behavioural effects of brain lesions', *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* **355**(1393), 147–161. Publisher: Royal Society. doi: 10.1098/rstb.2000.0555 (pages 28, 46).

Yu, Q., Du, Y., Chen, J., Sui, J., Adali, T., Pearlson, G. and Calhoun, V. D. (2018), 'Application of Graph Theory to Assess Static and Dynamic Brain Connectivity: Approaches for Building Brain Graphs', *Proceedings of the IEEE. Institute of Electrical and Electronics Engineers* **106**(5), 886–906. doi: 10.1109/JPROC.2018.2825200 (page 26).

Zador, A. M. (2019), 'A critique of pure learning and what artificial neural networks can learn from animal brains', *Nature Communications* **10**(1), 3770. Bandiera_abtest: a Cc_license_type: cc_by Cg_type: Nature Research Journals Number: 1 Primary_atype: Reviews Publisher: Nature Publishing Group Subject_term: Computer science;Network models Subject_term_id: computer-science;network-models. doi: 10.1038/s41467-019-11786-6 (page 37).

Zhang, C., Raghu, M., Kleinberg, J. and Bengio, S. (2022), 'Pointer Value Retrieval: A new benchmark for understanding the limits of neural network generalization'. arXiv:2107.12580. http://arxiv.org/abs/2107.12580 (accessed on: 2024-12-02) (pages 67, 71).

Zhang, J., Abiose, O., Katsumi, Y., Touroutoglou, A., Dickerson, B. C. and Barrett, L. F. (2019), 'Intrinsic Functional Connectivity is Organized as Three Interdependent Gradients', *Scientific Reports* **9**(1), 15976. Publisher: Nature Publishing Group. doi: 10.1038/s41598-019-51793-7 (page 26).

Zhang, X.-J., Moore, J. M., Gao, T.-T., Zhang, X. and Yan, G. (2025), 'Brain-inspired wiring economics for artificial neural networks', *PNAS Nexus* **4**(1), pgae580. doi: 10.1093/pnasnexus/pgae580 (pages 36, 69, 77, 88).

Zilles, K. (2018), 'Brodmann: a pioneer of human brain mapping—his impact on concepts of cortical organization', *Brain* **141**(11), 3262–3278. doi: 10.1093/brain/awy273 (pages 24, 44).