AUTONOMOUS MEMORY REHEARSAL IN ASSOCIATIVE MEMORY NETWORKS AND ITS IMPLICATIONS FOR BIOLOGICALLY PLAUSIBLE CONTINUOUS LEARNING

Paul Saighi, Marcelo Rozenberg
Laboratoire de Physique des Solides
CNRS, Université Paris-Saclay
Orsay, 91405, France
paul.saighi@universite-paris-saclay.fr

ABSTRACT

The brain's faculty to assimilate and retain information, continually updating its memory while limiting the loss of valuable past information, remains largely a mystery. We address this challenge related to continuous learning in the context of associative memory networks, where the iterative storage of correlated patterns traditionally requires non-local learning rules or external memory systems. Our work demonstrates how incorporating biologically-inspired inhibitory plasticity enables networks to autonomously explore their attractor landscape through local dynamics alone. The autonomous recovery of stored patterns enables continuous addition of new memories by allowing them to be incorporated with previously stored information while limiting its degradation, drawing parallels with memory consolidation during sleep-like states in biological systems. The resulting framework provides insights into how neural circuits might maintain memories through purely local interactions, while suggesting new approaches for building more biologically plausible approaches to continuous learning.

1 INTRODUCTION

Here, we address the issue of autonomous memory rehearsal and its application for Continuous Learning (CL) in associative-memory networks. For the sake of bio-plausibility and potential implementation in neuromorphic substrates, we shall demand that our system exhibits the following features: 1) It incorporates memory states that can be correlated; 2) The network is able to converge into the stored states while limiting the visit of spurious patterns, i.e., strange attractors corresponding to none of the stored memories; 3) Plasticity rules have to be local, meaning that the modification of synaptic efficacy depends solely on states that can be computed and shared by both postsynaptic and presynaptic neurons; 4) It should be autonomous, namely, it cannot recall an external list of previously recorded states or even rely on external cues to experience specific memory replay.

For requirements (1) and (3), we shall adopt a perceptron-like algorithm, inspired by Diederich & Opper (1987). Requirement (2) has motivated the use of continuous Hopfield networks (CHNs) (Hopfield, 1984). As long as we remain under a certain load, CHNs allow us to converge exclusively toward stored patterns, even for small networks and without using partial cues to help the convergence. This behavior is in contrast to the presence of spurious states in discrete Hopfield networks (DHNs) and avoids the difficulties induced by stochastic Hopfield networks (SHNs) for which the temperature parameter has to be carefully chosen (Amit et al., 1985).

Finally, our main contribution is to introduce an algorithm to address requirement (4). We shall show how a mechanism inspired by spontaneous neural activity during sleep allows one to autonomously recall and maintain the set of previously recorded memory states (Robins, 1995; Louie & Wilson, 2001; Peyrache et al., 2009; Fauth & Van Rossum, 2019; Tononi & Cirelli, 2014). A crucial feature of our approach is the use of inhibitory recurrent plastic synapses that allow for a sequential

and thorough search of all previously stored correlated memory states. The presence of inhibitory synapses that undergo Hebbian plasticity to shrink the attraction basin of previously visited attractor states is strongly inspired by computational neuroscience works based on actual neurophysiological data (Vogels et al., 2011).

2 MODEL

2.1 CONTINUOUS HOPFIELD NETWORK (CHN)

In contrast to the conventional HN model (Hopfield, 1982), where neural states are defined by discrete (binary) vectors, in a CHN each neuron of the network has its activity encoded by a continuous variable. The dynamic of the network is therefore described by a set of differential equations with each unit operating a leaky integration:

$$c\frac{du_i}{dt} = \sum_j W_{ij} v_j - \frac{u_i}{r} \tag{1}$$

where u_i is the membrane potential of neuron i, c is the membrane capacitance, r is the leak resistance of each neuron, W_{ij} is the synaptic efficacy between neurons j and i, and v_i is the activity (or firing rate) of neuron i that depends solely on the potential as

$$v_i = \sigma(u_i)$$

where σ is a monotonically increasing function of u with saturation to prevent runaway dynamics. In our case, $\sigma(x) = \frac{1}{1+exp(-x)}$. As $\sigma(0) = 0.5$, each unit has a positive output at its resting state, allowing the network to have activity without external current input.

The convergence on stable states for symmetric synaptic weights W_{ij} has been demonstrated (Hopfield, 1984). Throughout this work, whenever we integrate these equations using Euler method, we do so until the network reaches convergence, defined as $|\frac{du}{dt}|_\infty < \epsilon$, where $\epsilon = 10^{-6}$.

2.2 PATTERN STORAGE

CHNs evolve on $[0, 1]^n$ through continuous dynamics with n the number of neurons, allowing any state in this space to represent a stored pattern. For simplicity, we restrict ourselves to store binary patterns as stable states for which active neurons have a large firing rate, $v_i \approx 1$, and inactive neurons are nearly inactive, $v_i \approx 0$. We therefore use a threshold to interpret the network's output when needed (A.1). It is important to note that we do not take advantage of the continuous nature of our model to store patterns through a greater variety of states. The interest of CHNs is the lack of spurious states encountered under a critical load.

For a binary pattern $\boldsymbol{x}^\mu$, we define target potentials $\tilde{\boldsymbol{u}}$ as:

$$\tilde{\boldsymbol{u}}_i = \begin{cases} +u_{\text{target}} & \text{if } x_i^\mu = 1 \\ -u_{\text{target}} & \text{if } x_i^\mu = 0 \end{cases}$$

where u_{target} is chosen such that $\sigma^{-1}(-u_{\text{target}}) < \sigma^{-1}(v_{\text{thr}}) < \sigma^{-1}(u_{\text{target}})$, ensuring proper pattern reading after the thresholding procedure.

We introduce a perceptron-inspired learning algorithm for efficient pattern storage in the CHN (Diederich & Opper, 1987). Alg. 1 minimizes the error between desired target states and the network's equilibrium states, ensuring each memory becomes a stable attractor that can be retrieved via partial cues or rehearsal. Derivation of the weight update rule can be found in the Appendix (A.4).

Although a rigorous proof of convergence for Alg. 1 is beyond the scope of the present work, it is reasonable to expect that the arguments used to demonstrate the convergence of the gradient descent algorithm (GDA) in discrete HNs (DHN) could be adapted for this purpose (Diederich & Opper, 1987). Here we shall be content with numerical evidence demonstrating the network's ability to successfully query and revisit stored patterns. The querying procedure is indicated in the Appendix (A.2).

Algorithm 1 Gradient descent for the storage of correlated patterns

Initialize $W_{ij} = 0$ for all i, j
repeat
  for each pattern μ **do**
    Set target potentials: $\tilde{u}_i^\mu = +u_{\text{target}}$ if $x_i^\mu = 1$, else $-u_{\text{target}}$
    Compute current potentials: $\hat{u}_i^\mu = r \sum_j W_{ij} \sigma(\hat{u}_j^\mu)$
    Update weights: $W_{ij} \leftarrow W_{ij} - \alpha(\tilde{u}_i^\mu - \hat{u}_i^\mu) r \sigma(\hat{u}_j^\mu)$
  end for
until $\|\Delta \boldsymbol{W}\|_\infty < \epsilon$

2.3 Continuous incorporation of correlated memories

While the GDA effectively enables the storage of correlated memories, its implementation in Alg. 1 reveals a significant limitation: it requires multiple iterations over the entire pattern corpus to achieve convergence. By repeatedly processing all patterns, the algorithm can find a weight matrix that properly separates correlated patterns (Diederich & Opper, 1987).

Since finding appropriate weights requires processing all patterns concurrently, adding a new pattern would require access to all previously stored patterns from an external source. This requirement for external access to the complete training dataset stands in contrast to biological learning systems, which must incorporate new information while maintaining past memories without maintaining an explicit external copy of the original data. To overcome this external dependency and move towards more biologically plausible learning, we need to develop a mechanism that allows the network to internally recover its stored patterns. The development of such an autonomous pattern retrieval mechanism would allow us to harness the GDA's properties for continuous incorporation of correlated memories, as stored patterns can be internally recovered and combined with new inputs during the learning process (Alg. 2). Without such a mechanism to retrieve and reprocess all patterns, the network would suffer from catastrophic forgetting (also called catastrophic interference), where learning a new pattern in isolation rapidly erodes previously stored memories (McCloskey & Cohen, 1989; Robins, 1995; Kirkpatrick et al., 2017).

Algorithm 2 Continuous incorporation of correlated patterns through autonomous retrieval

Input: CHN trained on p patterns using the GDA
Given: New pattern $\boldsymbol{x}^{p+1}$ to be stored
Retrieve set $\{\boldsymbol{x}\}$ of stored patterns through autonomous rehearsal
Update pattern set: $\{\boldsymbol{x}\} \leftarrow \{\boldsymbol{x}\} \cup \{\boldsymbol{x}^{p+1}\}$
Apply GDA to store updated pattern set

This approach illustrates a more general principle: when a network has access to both a new memory trace $\boldsymbol{x}^{p+1}$ and previously stored traces $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^p$, it can perform operations analogous to the GDA, finding optimal synaptic configurations that leverage their shared characteristics, such as ensuring their separation.

2.4 Autonomous rehearsal

Given a trained network, we introduce a mechanism for autonomous pattern retrieval. When initialized to the neutral state ($u_i(t = 0) = 0$), the network dynamics described by Eq. 1 converges deterministically to a stored attractor. Although adding noise to these dynamics was explored as a potential solution for visiting multiple attractors, it proved ineffective for systematic pattern retrieval. Instead, we introduce a second set of plastic inhibitory synapses W'_{ij}:

$$c\frac{du_i}{dt} = \sum_j W_{ij} v_j - \sum_j W'_{ij} v_j - \frac{u_i}{r} \tag{2}$$

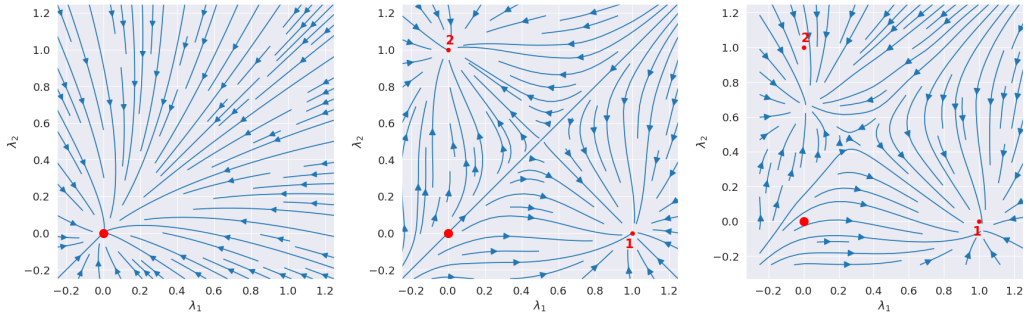
Each visited attractor is made shallower by the Hebbian-like plasticity of the inhibitory synapses, following the procedure detailed in Alg. 3. Fig. 2a provides a visualization using binary-digit pat-

terns from the MNIST dataset (Deng, 2012). As shown in Fig. 1, modifying inhibitory synaptic weights reshapes the attraction basins of the network, allowing sequential exploration of stored patterns. During quiescent states such as sleep or rest, the mammalian central nervous system exhibits a similar memory rehearsal dynamic. This process is believed to function as a consolidation mechanism that mitigates or modulates memory forgetting (Robins, 1995; Louie & Wilson, 2001; Peyrache et al., 2009; Fauth & Van Rossum, 2019; Tononi & Cirelli, 2014).

Say that this state where the network autonomously revisit stored patterns in an offline mode to then consolidate memories is inspired by sleep, then put ref

Algorithm 3 Autonomous Rehearsal

Input: Trained network weights W_{ij}, iterations k, plasticity rate β
Initialize: $W'_{ij} = 0$, $\{\boldsymbol{x}\} = \emptyset$
while $j < k$ **do**
    Set neutral initial conditions: $u_i(t = 0) = 0$
    Biased phase: Integrate Eq. 2 until convergence to bias dynamics away from visited patterns
    Free phase: Integrate Eq. 1 until convergence to complete pattern retrieval
    Read pattern $\boldsymbol{x}^\mu$ from final state via thresholding as introduced Sec. 2.2
    Update pattern set: $\{\boldsymbol{x}\} \leftarrow \{\boldsymbol{x}\} \cup \{\boldsymbol{x}^\mu\}$
    Update inhibitory weights: $\Delta W'_{ij} = \beta v_i(t_f) v_j(t_f)$ with $v_i(t_f)$ the rate of neuron i at the end of the free phase.
    $j \leftarrow j + 1$
end while
Return: $\{\boldsymbol{x}\}$

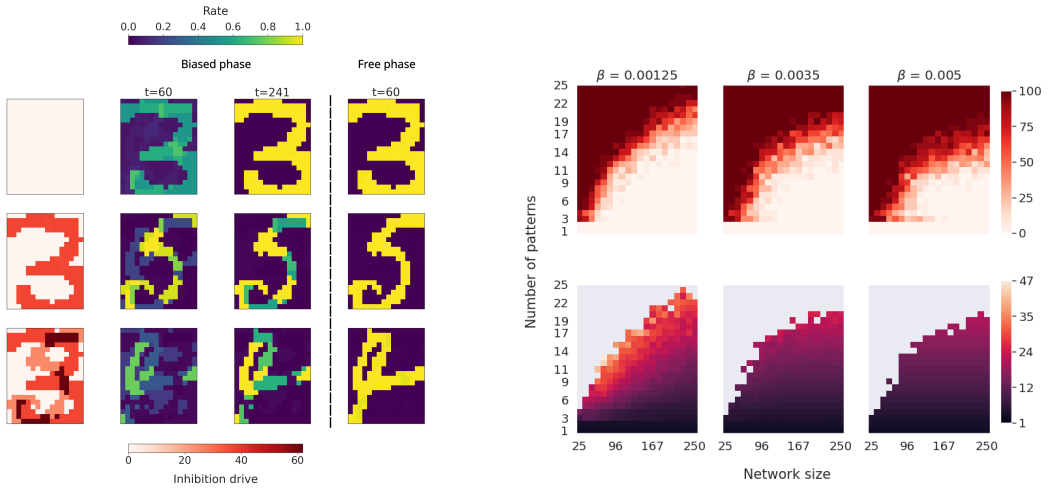


(a) Pre-training field (b) Post-training field (c) Post-inhibition field, $\beta = 0.2$

Figure 1: **Stream plots of derivative for a CHN in a 2D subspace spanned by two pattern states.** We consider two stable states $\mathbf{u}^1$ and $\mathbf{u}^2$ corresponding to two patterns; these define a two-dimensional subspace parametrized by $\mathbf{u}(\lambda_1, \lambda_2) = \lambda_1 \mathbf{u}^1 + \lambda_2 \mathbf{u}^2$. At each point $\mathbf{u}(\lambda_1, \lambda_2)$ in this subspace, we compute the full N-dimensional time derivative via Eq. 2 and project it back onto the $\{\mathbf{u}^1, \mathbf{u}^2\}$-plane to obtain the plotted flow. (a) Before training. (b) After storing the two patterns using the GDA, each pattern state (small red dots) becomes a stable attractor; the neutral state $\mathbf{u}_N = \mathbf{0}$ (large red dot) is on an unstable manifold. (c) Inhibitory potentiation of the second pattern "shrinks" its basin of attraction and shifts the separatrix. This observed geometry explains why even a small change in the inhibitory weights can drastically alter the trajectory from the neutral initial condition, driving the network toward a different attractor. We can observe that the network doesn't converge precisely toward the stored state anymore, but slightly aside. As this distortion is minimal for small β, it is largely compensated by the thresholding mechanism used to read the network output (A.1). The free phase described in 3 further reduces the effect of this distortion by allowing the network to converge to the 'true' stable state after a biased phase that leverages the shift in the separatrix. Nevertheless, this distortion accumulates with each visited pattern, which can still cause the emergence of spurious patterns when attempting to retrieve too many patterns from the network (Fig. 2b).

3 PRELIMINARY RESULTS

As demonstrated in Fig. 2b, our autonomous rehearsal (AR) mechanism successfully achieves the key objective outlined in the introduction: enabling networks to systematically recover stored patterns without external cues or memory lists. This autonomous exploration of the attractor landscape scales with network size - larger networks can reliably recover more stored patterns. This allows the implementation of continuous pattern incorporation as described in Alg. 2. The values of β must be relatively small compared to the target potential u_{target}, in our case smaller than 0.005. This allows for gentle nudging of the network during recovery while limiting the appearance of spurious patterns induced by an important deformation of the energy landscape.



(a) (b)

Figure 2: **(a)** Recovery of three correlated patterns in a network of 20×16 units. Left panels (in red): the evolution of the inhibitory drive to each neuron, defined as $i_j^{inh} = \sum_i W'_{ij}$. Right panels (blue and yellow): evolution of the rates v_i in time for each unit of the network. Each row corresponds to the start of a new simulation of the CHN. The black dashed line bar corresponds to the removal of inhibitory weights, allowing the convergence to a stored pattern after the biased recollection as described in Alg. 3. An exaggerated inhibitory potentiation $\beta = 0.25$ is used to illustrate the stable states deformation at the end of the biased phase for which the free phase is necessary. **(b)** Recovery capacity of AR for various β. Correlated patterns are generated with a ρ of 0.5 using Alg. 5 (A.3). For networks of various sizes and for various number of stored patterns, we use Alg. 3 until all patterns are recovered or until we find a spurious state. Data are averaged over 20 simulations for various pattern corpora. (Top row) The percentage of simulations where at least one spurious pattern occurs before recovering all patterns. (Bottom row) Number of iterations before recovering all patterns. Lower values of β require more iterations to recover the whole corpus of patterns. The higher the value of β, the higher the chances to find spurious patterns as the number of stored patterns increase. Larger values of β dramatically impair the recovery dynamics.

4 CONCLUSION

In this work, we demonstrate how inhibitory synaptic plasticity enables autonomous exploration of attractor landscapes in continuous Hopfield networks. Our key finding reveals that inhibitory plasticity allows networks to systematically visit different stored patterns guided solely by their activity history, even for strongly correlated memories. The capacity for self-directed pattern exploration, emerging from the use of a gradient descent algorithm and inhibitory modulation, offers insights for both biological memory consolidation and neuromorphic computing. Moreover, this autonomous rehearsal mechanism and the gradient descent algorithm together allow the incorporation of new memories while helping to preserve existing ones, contributing to the ongoing challenge of mitigating catastrophic forgetting in neural networks.

REFERENCES

Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Storing Infinite Numbers of Patterns in a Spin-Glass Model of Neural Networks. *Physical Review Letters*, 55(14):1530–1533, September 1985. ISSN 0031-9007. doi: 10.1103/PhysRevLett.55.1530. URL https://link.aps.org/doi/10.1103/PhysRevLett.55.1530.

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

Sigurd Diederich and Manfred Opper. Learning of correlated patterns in spin-glass networks by local learning rules. *Physical Review Letters*, 58(9):949–952, March 1987. ISSN 0031-9007. doi: 10.1103/PhysRevLett.58.949. URL https://link.aps.org/doi/10.1103/PhysRevLett.58.949.

Michael Jan Fauth and Mark Cw Van Rossum. Self-organized reactivation maintains and reinforces memories despite synaptic turnover. *eLife*, 8:e43717, May 2019. ISSN 2050-084X. doi: 10.7554/eLife.43717. URL https://elifesciences.org/articles/43717.

J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, April 1982. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.79.8.2554. URL https://pnas.org/doi/full/10.1073/pnas.79.8.2554.

J J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81(10):3088–3092, May 1984. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.81.10.3088. URL https://pnas.org/doi/full/10.1073/pnas.81.10.3088.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, March 2017. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1611835114. URL https://pnas.org/doi/full/10.1073/pnas.1611835114.

Kenway Louie and Matthew A. Wilson. Temporally Structured Replay of Awake Hippocampal Ensemble Activity during Rapid Eye Movement Sleep. *Neuron*, 29(1):145–156, January 2001. ISSN 08966273. doi: 10.1016/S0896-6273(01)00186-6. URL https://linkinghub.elsevier.com/retrieve/pii/S0896627301001866.

Michael McCloskey and Neal J. Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. Elsevier, 1989. ISBN 978-0-12-543324-2. doi: 10.1016/S0079-7421(08)60536-8. URL https://linkinghub.elsevier.com/retrieve/pii/S0079742108605368.

Adrien Peyrache, Mehdi Khamassi, Karim Benchenane, Sidney I Wiener, and Francesco P Battaglia. Replay of rule-learning related neural patterns in the prefrontal cortex during sleep. *Nature Neuroscience*, 12(7):919–926, July 2009. ISSN 1097-6256, 1546-1726. doi: 10.1038/nn.2337. URL https://www.nature.com/articles/nn.2337.

Anthony Robins. Catastrophic Forgetting, Rehearsal and Pseudorehearsal. *Connection Science*, 7(2):123–146, June 1995. ISSN 0954-0091, 1360-0494. doi: 10.1080/09540099550039318. URL https://www.tandfonline.com/doi/full/10.1080/09540099550039318.

Giulio Tononi and Chiara Cirelli. Sleep and the Price of Plasticity: From Synaptic and Cellular Homeostasis to Memory Consolidation and Integration. *Neuron*, 81(1):12–34, January 2014. ISSN 08966273. doi: 10.1016/j.neuron.2013.12.025. URL https://linkinghub.elsevier.com/retrieve/pii/S0896627313011860.

T. P. Vogels, H. Sprekeler, F. Zenke, C. Clopath, and W. Gerstner. Inhibitory Plasticity Balances Excitation and Inhibition in Sensory Pathways and Memory Networks. *Science*, 334(6062):1569–1573, December 2011. ISSN 0036-8075, 1095-9203. doi: 10.1126/science.1211095. URL https://www.science.org/doi/10.1126/science.1211095.

A APPENDIX

A.1 READING NETWORK'S OUTPUT

A pattern $\boldsymbol{x}^\mu$ is defined as a binary vector such that $\boldsymbol{x}^\mu = (x_1^\mu, x_2^\mu, \ldots, x_N^\mu)$ where $x_i^\mu \in \{0, 1\}$ for each unit i. A pattern is read from the state of the network at time t using a threshold:

$$x_i^\mu = \begin{cases} 1 & \text{if } v(t)_i > v_{thr} \\ 0 & \text{otherwise} \end{cases}$$

with $v_{thr} = 0.5$ in our case.

A.2 QUERYING

The network's ability to retrieve stored patterns can be tested through partial cues. Given a subset of informed units, we initialize their potentials according to the partial pattern while leaving other units at rest:

$$u_i(t=0) = \begin{cases} +u_{\text{target}} & \text{if unit } i \text{ is informed and } x_i = 1 \\ -u_{\text{target}} & \text{if unit } i \text{ is informed and } x_i = 0 \\ 0 & \text{otherwise} \end{cases}$$

Algorithm 4 Querying with convergence of the network using Euler method

set $u_i(t=0) = \begin{cases} u_{\text{up}} & \text{if } x_i^\mu = 1 \\ u_{\text{down}} & \text{if } x_i^\mu = 0 \\ 0 & \text{if } x_i^\mu \text{ not informed} \end{cases}$
repeat
  for each unit i **do**
    $u_i(t \mathrel{+}= 1) = \delta(\sum_j W_{ij} v_j - \frac{u_i(t)}{r})$
  end for
until $\|\dot{\boldsymbol{u}}\|_\infty < \epsilon$

The network then evolves according to Eq. 1 until reaching a stable state ($\|\frac{d\boldsymbol{u}}{dt}\|_\infty < \epsilon$). When operating below capacity, the dynamics typically converge to the stored pattern most similar to the initial cue. The final state is interpreted as a binary pattern using the thresholding procedure. As with traditional DHNs, the retrieval success depends on both the network load and the number of informed units in the initial cue. While a theoretical analysis of the storage capacity under Alg. 1 would be of interest, our focus here is on the autonomous rehearsal mechanism, which operates in a regime well below this limit where pattern retrieval is highly reliable.

A.3 CONSTRUCTION OF CORRELATED PATTERNS

Algorithm 5 Generation of p correlated random patterns

Generate a random binary pattern $\boldsymbol{x}^{parent} \in \{-1, 1\}^N$
Set $k = \lfloor (1-\rho)N \rfloor$, where ρ is the ratio of bits not randomized
for $i = 1$ to p **do**
  $\boldsymbol{x}^i \leftarrow \boldsymbol{x}^{\text{parent}}$
  Randomly select k distinct indices $\{j_1, \ldots, j_k\}$ from $\{1, \ldots, N\}$
  for $m = 1$ to k **do**
    $x_{j_m}^i \leftarrow$ random choice from $\{-1, 1\}$ with uniform distribution
  end for
end for

A.4 GRADIENT DESCENT FOR PATTERN STORAGE

At steady state, when $\frac{du_i}{dt} = 0$, from Eq. 1 we have:

$$\hat{u}_i = r \sum_j W_{ij} v_j$$

where $v_j = \sigma(\hat{u}_j)$.

We can then define the error function E as the sum of squared differences between steady-state potentials $\hat{u}_i$ and target potentials $\tilde{u}_i$:

$$E = \frac{1}{2} \sum_i (\hat{u}_i - \tilde{u}_i)^2$$

Taking the derivative with respect to the steady-state potential:

$$\frac{\partial E}{\partial \hat{u}_i} = (\hat{u}_i - \tilde{u}_i)$$

To compute how the error changes with respect to the weights W_{ij}, we use the chain rule:

$$\frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial \hat{u}_i} \frac{\partial \hat{u}_i}{\partial W_{ij}}$$

The partial derivative with respect to W_{ij} is:

$$\frac{\partial \hat{u}_i}{\partial W_{ij}} = r v_j$$

This is an approximation that captures the immediate direct effect of W_{ij} on $\hat{u}_i$, assuming v_j remains constant and ignoring feedback effects in the recurrent network. This approximation is valid when optimization steps are small relative to the nonlinearities in the sigmoid function.

Therefore, the error gradient with respect to W_{ij} becomes:

$$\frac{\partial E}{\partial W_{ij}} = (\hat{u}_i - \tilde{u}_i) r v_j$$

Finally, we update the weights using gradient descent:

$$\Delta W_{ij} = -\alpha \frac{\partial E}{\partial W_{ij}} = -\alpha r (\hat{u}_i - \tilde{u}_i) v_j$$

where α is the learning rate, typically set to 0.001 to ensure convergence.

A.5 PARAMETERS

Table 1: CHN parameters

Name	Value	Description
ϵ_{sim}	10^{-6}	Convergence constant
r	1	Leak time constant
c	1	Integration time constant
δ	0.001	Euler step size

Table 2: Gradient descent

Name	Value	Description
α	0.0001	Learning rate
ϵ_{learn}	10^{-6}	Convergence constant
u^{target}	6	Target potential winning units