Reconstructing creative thoughts: Hopfield Neural Networks

Denisa Checiu[1], Mathias Bode[1] and Radwa Khalil[2*]

[1]School of Electrical and Computer Engineering, Constructor University, Bremen, Germany

[2]School of Business, Social, and Decision Sciences, Constructor University, Bremen, Germany

Author Note

[Radwa Khalil: https://orcid.org/0000-0003-2632-8306]

Corresponding Author: rkhalil@constructor.university

Abstract

From a brain processing perspective, the perception of creative thinking is rooted in the underlying cognitive process, which facilitates exploring and cultivating novel avenues and problem-solving strategies. However, it is challenging to emulate the intricate complexity of how the human brain presents a novel way to uncover unique solutions. One potential approach to mitigate this complexity is incorporating creative cognition within the evolving artificial intelligence systems and associated neural models. Hopfield Neural Networks (HNN) are commonly acknowledged as a simplified neural model, renowned for their biological plausibility to store and retrieve information, specifically patterns of neurons. Therefore, we propose for the first time using HNN to generate a potential model that emulates specific cognitive processes of creative thinking based on generating meaningful links between seemingly disparate concepts.

***Keywords*:** Creative Thinking, Hopfield Neural Network, Patterns, Memory, Associative Chains

Introduction

Recent years have witnessed substantial advances and tremendous growth in artificial intelligence (AI) systems and their applications (Kappas & Gratch, 2023; Virvou, 2022; Zhang et al., 2022). The groundbreaking progress of conversational AI such as ChatGPT, which broke significant barriers of automation in communication (Abdullah et al., 2022; Wang et al., 2023), and the emerging era of human-AI interactions, in which our computers act as autonomous agents capable of making decisions (Kappas & Gratch, 2023; Virvou, 2022), is indisputable.

AI research can be classified into two primary categories: the emulation category, which seeks to understand human perception, cognition, and motor abilities (e.g., humanoid robots), and the application category, which seeks to use AI methods to model and create large-scale products (e.g., teleoperated devices)(Kappas & Gratch, 2023; Shneiderman, 2020).

This article focuses on emulation because we aimed to employ AI to study creative cognition. Through the creative act, we can perceive the world differently, find hidden patterns, and construct connections between seemingly unrelated events to generate extraordinary outputs (Abraham, 2018; Herrmann, 1981; Khalil & Demarin, 2023; Khalil & Moustafa, 2022). Mendelson's (1974) model of knowledge access employed the concepts of defocused and focused attention to elucidate the semantic hierarchy of creative connections. The concept of the associative foundation of the creative process aligns with the earlier introduction of this notion by Mednick (1962).

It is widely accepted that creative products should express novel, valuable, and may be surprising ideas (Corazza, 2016; Gardner, 1994; Runco & Jaeger, 2012; Torrance, 1974; Wilson et al., 1954). Consequently, there is a broad consensus that creative thinking is

recognized for its ability to generate solutions that are both novel, useful, and surprising (Han et al., 2021; Khalil & Moustafa, 2022; Mastria et al., 2022).

Nevertheless, creativity is multifaceted: it can be expressed in various forms, hues, and shapes and is, thus, enormously challenging to pin down (Abraham, 2013; Gaut, 2010; Kaufman & Beghetto, 2009; Sawyer, 2006, 2011; Squalli & Wilson, 2014; Tardif & Sternberg, 1988). One possibility to reduce this complexity and overlapping is to focus on the neural network architectures and their relevant learning rules, as Khalil and Moustafa (2022) suggested.

Therefore, we intend to incorporate this creative cognitive process-based semantic association (Beaty & Kenett, 2023; Kenett & Beaty, 2023; Li et al., 2021) into a conceptual framework for a creative, plausible neural network system using the Hopfield neural network (HNN). The biologically inspired mathematical tool HNN, developed by John J. Hopfield in 1982, is best recognized for its ability to store and recall knowledge (McEliece et al., 1987). Establishing the HNN model was remarkable in the neural computational and modeling era (Abe, 1989; Hopfield, 1982; KÖksal & Sivasundaram, 1993; Ramsauer et al., 2021).

HNN is a recurrent associative neural network model that is featured by having input from all neurons, and thus, it can retrieve memories from partial or noisy input (Hopfield, 1982). After receiving an input, the HNN uses iterative updating to update the neuron states synchronously or asynchronously to converge toward one of the stored memories and reach a stable state that recalls the information it was trained (Krotov & Hopfield, 2016). Therefore, HNN has been used for associative memory tasks, including associating input with its most similar memorized pattern or identifying unknown systems by computing the learning mechanism's optimized coefficients to fit the nonlinear system (McEliece et al., 1987; Wang et al., 2003, 2005).

Using this network structure strongly resembles the architecture of human memory, which encodes several neurons that fire together in response to certain "microfeatures"(Gabora, 2010). In other words, the HNN recalls patterns by considering the correlation between the neurons using the weights matrix, not by storing the patterns; thus, we chose the HNN model to develop a semantic association-based creative conceptual framework (Beaty et al., 2023; Kenett & Beaty, 2023; Luchini et al., 2023).

We describe this semantic linkage as a radio knob that we can turn to tune whether the network can solve problems creatively or if it shuts down entirely; the threshold is used as a parameter. We refer to it as the "**first knob of creativity"**; in this context, we are invoking the "**second knob of creativity"** to facilitate exploring alternative solutions inside our network. By manipulating the knob, it is possible to selectively suppress specific patterns, facilitating the creative functioning of the HNN and identifying other patterns with which our input can be linked.

Methods

This section describes the neural network structure and implementation of HNN (Fig.1).

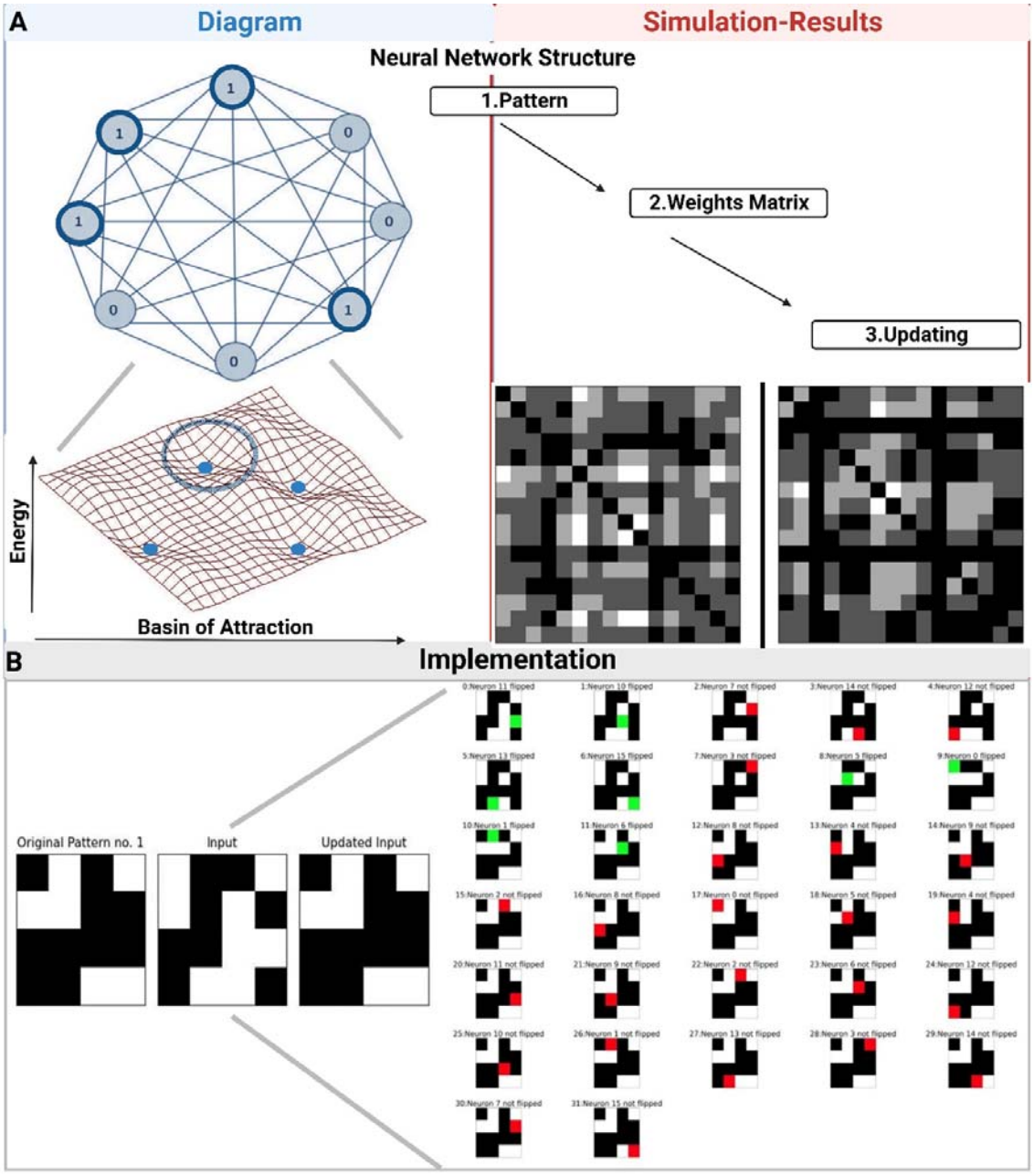

Figure 1. Description of the neural network structure of Hopfield neural networks (HNN) and its implementation. Panel A depicts a schematic representation of a binary HNN, illustrating its connectivity and the potential states of the neurons, which can be either on (active) or off (inactive). The lower graph depicted in panel A refers to the energy structure of HNN with multiple point attractors; fixed points are shown as small blue dots, and a dotted circle denotes a basin of attraction. The network consists of a pattern and a weight matrix [W], which directs its capacity to undergo single-step training as the computation is executed in a singular operation. The diagonal symmetry of W can be attributed to the symmetric and bi-directional connections among the

neurons, which is a consequence of the complete connectivity. The lower graph of panel A on the right side illustrates visual representations of the weight matrix obtained from preliminary experiments conducted on binary HNN implementation. Panel B illustrates the graphical visualizations of the successful implementation of HNN. The graph on the left-hand side shows a randomly chosen pattern, its perturbation as the input fed into the HNN, and the output after two complete cycles of asynchronous updates from the initial experiments. The graph on the right-hand side represents asynchronous neuron updates of an input iterated through the network - a green cell means the neuron was updated, while a red cell translates to it staying the same value.

Neural Network Structure

The network structure includes patterns and weight matrix updating (Fig.1a).

1. Patterns

We define n patterns, each length N; every pattern pi, with i = [1, n], is considered a memory network, which will learn during the training process and later reconstruct given an input close to or derived from it, essentially an input resembling it (Fig.1a).

$$n = \text{no. of patterns}$$
$$N = \text{no. of neurons per pattern}$$
$$pi = \text{pattern i of size N } \forall i = [1,n]$$

A pattern pi comprises N neurons, every neuron having a state, on or off. We used a binary HNN; a neuron can be either 0, meaning the neuron is not firing, or 1, the neuron is firing. This binary mode reflects a simplistic and discrete view of how the neurons act (Šíma et al., 2000). Therefore, for a general definition, we can consider each pattern as a random combination of 0s and 1s, each representing a neuron, forming a row vector.

$$pi = [0,0,1,0,\ldots,1,0,1] \ \forall i = [1,n]$$

Each neuron is connected to all the others from the network, indicating that each connection has a unique weight, similar to the synapses between the neurons, representing communication between neurons and how these change their states accordingly (Ramsauer et

al., 2021). An ensemble of specific neurons from different areas firing together in a unique sequence refers to a memorized concept (Hopfield, 1982).

We added another variable that can control the number of ones we have in all patterns - thus, we ensure that patterns are still randomly generated, but we know how many neurons are firing. We started by generating patterns with 10% of ones out of the total of N neurons (N = 1024). Given a noisy input, we were ready to perform a more thorough analysis of the range of thresholds for each n for which the network can adequately reconstruct patterns.

We performed 50 tests for each possible threshold value, ranging from 0 to 150 with increments of 10, for each number of patterns n, starting from 1 until 9. A test consisted of generating n new patterns, which contained 10% of randomly firing neurons, and then perturbing by 100 neurons a randomly chosen pattern, which we fed into the network as input.

Our goal is to develop a neurocomputational model for creative thinking based on semantic associations (Beaty et al., 2023; Benedek et al., 2023; Gabora, 2010; Gerver et al., 2023); it is logical to have less correlated concepts to identify the unexplored connections in between. Based on this reasoning, our patterns represent unrelated concepts; therefore, we do not require a large percentage of neurons firing per pattern (i.e., meaning less likelihood for correlation)(Amari & Maginu, 1988; Anishchenko & Treves, 2006; Hopfield, 1982; McEliece et al., 1987).

2. Weights Matrix

We structure the connectivity of the neurons as a matrix with defined weights. We included *n* patterns in our HNN to store and memorize; this process is performed through the computation of the weights matrix *W*.

We used the Hebbian learning rule to store the patterns in HNN. The fundamental concept underlying the Hebbian rule is encapsulated in the widely recognized phrase,**" Neurons that fire together wire together. Neurons that fire out of sync fail to link"**, meaning if two neurons fire simultaneously, their connectivity value - the unique weight connecting them - increases, and in return, causes their activity to become more correlated over time (Morris, 1999).

The reasoning above may be applied to the alternative scenario, wherein the neurons exhibit contrasting states. In this situation, their connection value diminishes and reaches a null state, specifically in our context. This tendency facilitates the convergence of inputs towards the most similar pattern, signifying the synchronized firing of neurons and indicating the connection between them (Amit, 1995; Brunel, 1996).

Based on this Hebbian rule, we created W; thus, the weights matrix is computed by summing up the outer product of each pattern p_i - to measure the correlation between the patterns and our input.

$$W = \sum_{i=1}^{n} p_i^T \cdot p_i = p_1^T \cdot p_1 + p_2^T \cdot p_2 + \cdots + p_n^T \cdot p_n$$

We considered each pattern p_i a row vector, and as such p^T_i is a column vector, the outer product computation of a pattern resulting in a matrix.

One of the remarkable advantages of the HNN is its ability to train in one step, as the computation of the weights matrix W is completed in one operation (Abe, 1989; Abe & Gee, 1995; Atencia et al., 2005; Joya et al., 2002; KÖksal & Sivasundaram, 1993; Ramsauer et al., 2021). We used the diagonal symmetry of W, caused by the symmetric and bi-directional connections of the neurons, a direct result of the full connectivity (Fig.1a).

3. Updating

Given a binary input x of N neurons, we iterate it through HNN and make it converge towards a stable state. This convergence is facilitated by the utilization of a weights matrix W. Notably, the patterns themselves are not required for this operation, as the network has been trained, and all the necessary information is encapsulated within W (Ramsauer et al., 2021; Šíma et al., 2000).

There are two ways to perform an update: synchronously, meaning all of the neurons from the input x get updated in one iteration (Kobayashi, 2021; Michel et al., 1990), and asynchronously, where at each iteration, a random neuron (out of the N available ones) is chosen and updated (Aviel et al., 2004; Hopfield, 1982; Ramsauer et al., 2021). We used asynchronous updates to comprehensively examine the intermediate steps in reaching a solution. Consequently, an iteration is defined as updating a single input neuron, necessitating N iterations to fully update all input neurons.

Considering x as our input, W *as* the weights matrix, and t *as* a chosen threshold value, a synchronous update follows the formula:

$$\tilde{x} = sgn(x \cdot W - t)$$

When performing an asynchronous update of the input x at randomly chosen position j, we compute the dot product between the input x and column j of the weights matrix. In other words, we considered all the neurons in position j in the patterns and their respective connectivity value. Thus, the formula is:

$$\tilde{x}[j] = x \cdot W[1:n][j] = x \cdot [W(1,j), W(2,j), \dots W(n,j)]^T = \\ = (x_1, x_2, \dots x_n) \cdot [W(1,j), W(2,j), \dots W(n,j)]^T$$

The output of this formula will yield a positive numerical value with a minimum value of 0. This occurs when no correlations are observed between the firing neurons in the input and the patterns associated with the specific position j. The resulting values increase

proportionally to the strength of the correlation between the patterns, which directly influences the values in *W*. These values encapsulate the correlation between our input and the memorized patterns. Consequently, we applied a threshold to each value to convert it to 0 or 1.

$$x[j] = sgn(x \cdot W[1:n][j] - t) = \begin{cases} 1, & \text{if } (x \cdot W[1:n][j] - t) \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

To create a memorized pattern, it is necessary to minimize to reach a stable state, hence reaching a stable state, which is the main idea behind the updating process of HNN (Koiran, 1994; Ramsauer et al., 2021; Šíma et al., 2000).

–

Each stored pattern has a corresponding minimum energy, which acts as an attractor. Therefore, it is logical to keep minimizing the energy function, as these are the states we aim to achieve with each update (Koiran, 1994; Ramsauer et al., 2021; Šíma et al., 2000). The weights matrix *W* comprises the connectivity values that indicate a noticeable tendency to increase while storing similar patterns. Therefore, the energy function also has minimums corresponding to spurious states - they are not patterns but modifications or linear combinations, which can potentially cause our input to become trapped (Koiran, 1994; Šíma et al., 2000).

Implementation

The algorithmic implementation was conducted utilizing the Python programming language. All necessary features required for the network's functionality were developed from the ground up, incorporating several libraries like numpy, matplotlib, and random, among others (Davison, 2008). The patterns denoted by 'n' were generated by randomly combining

zeros and ones, resulting in a cumulative length of N neurons. The code availability is upon request.

1. Initial Experiments

The implementation process was initiated by constructing the system from its fundamental components, wherein various binary (i.e., 0s and 1s) and bipolar (i.e., 1s and -1s) patterns and inputs were explored through experimentation.

We selected the implementation of the binary HNN over the bipolar HNN because of its better alignment with the plausibility of neuron behavior, specifically in terms of firing when surpassing a specific activation threshold. The emphasis is placed on the magnitude of activation rather than the sign, as supported by previous studies (Litinskii, 2001; Weisbuch & Fogelman-Soulie, 1985). The preference for asynchronous neuron updates over synchronous updates is based on their advantage in facilitating the visualization of the network's iterative process in shifting an input towards a stable state, as opposed to only perceiving the result in a single step (Aviel et al., 2004; Hopfield, 1982).

During this phase, the network would often attempt to update a specific neuron; this is not only redundant, as it would yield the same outcome, but also an insufficient depiction of how neurons activated during the recall process.

Initially, we ensured that each input neuron was updated at least once - all input positions were randomized at the beginning of a complete cycle of neuron updates and later used to update the input asynchronously (Fig.1b). Thus, a full input update after N neuron iterations was ensured. This asynchronous updating approach was subsequently employed in all subsequent implementations.

Afterward, we introduced perturbations of HNN’s memorized patterns to determine the expected outcome after a series of iterations. The algorithm stochastically selects a single

pattern from the n patterns and perturbs its memory by flipping the state of some neurons, resulting in a change from 0 to 1 or vice versa. At first, the function was designed to accept a probability value representing the likelihood of each neuron being flipped. However, it was then determined that it would be more beneficial to specify the number of random neurons to be perturbed to gather data on the performance of the HNN. The patterns denoted as "n" were generated by randomly combining zeros and ones, resulting in a cumulative length of N neurons (Fig.1b).

After completing preliminary trials, we improved all the Python code and successfully developed a functional binary HNN. As a logical progression, we proceeded to gradually augment the number of patterns n and then analyzed the behavior of the HNN. It has been indicated that the memory capacity[1] of the HNN decreases as the overlap of patterns increases (Hopfield, 1982; McEliece et al., 1987; Ramsauer et al., 2021). Hence, before implementing the threshold, we rechecked the computation of the weights matrix W, specifically how the outer products of the patterns expand.

$$W = \sum_{i=1}^{n} p_i^T \cdot p_i = \sum_{i=1}^{n} \begin{bmatrix} \mathbf{p_1^{(i)}} \cdot p_1^{(i)}, \mathbf{p_1^{(i)}} \cdot p_2^{(i)}, \mathbf{p_1^{(i)}} \cdot p_3^{(i)}, \ldots \mathbf{p_1^{(i)}} \cdot p_N^{(i)} \\ \mathbf{p_2^{(i)}} \cdot p_1^{(i)}, \mathbf{p_2^{(i)}} \cdot p_2^{(1)}, \mathbf{p_2^{(i)}} \cdot p_3^{(i)}, \ldots \mathbf{p_2^{(i)}} \cdot p_N^{(i)} \\ \vdots \\ \mathbf{p_N^{(i)}} \cdot p_1^{(i)}, \mathbf{p_N^{(i)}} \cdot p_2^{(i)}, \mathbf{p_N^{(i)}} \cdot p_3^{(i)}, \ldots \mathbf{p_N^{(i)}} \cdot p_N^{(i)} \end{bmatrix}$$

We applied the unfolded formulation of *W* to our asynchronous update of neuron j.

$$\tilde{x}[j] = (x_1, x_2, \ldots, x_N) \cdot \begin{bmatrix} W_{(1,j)} \\ W_{(2,j)} \\ \vdots \\ W_{(n,j)} \end{bmatrix} = (x_1, x_2, \ldots, x_N) \cdot \begin{bmatrix} \sum_{i=1}^{n} p_1^{(i)} \cdot \mathbf{p_j^{(i)}} \\ \sum_{i=1}^{n} p_2^{(i)} \cdot \mathbf{p_j^{(i)}} \\ \vdots \\ \sum_{i=1}^{n} p_N^{(i)} \cdot \mathbf{p_j^{(i)}} \end{bmatrix} =$$

$$= x_1 \cdot \sum_{i=1}^{n} p_1^{(i)} \cdot \mathbf{p_j^{(i)}} + x_2 \cdot \sum_{i=1}^{n} p_2^{(i)} \cdot \mathbf{p_j^{(i)}} + x_N \cdot \sum_{i=1}^{n} p_N^{(i)} \cdot \mathbf{p_j^{(i)}}$$

[1] We refer t ich the inputs can reach a stab ield, 2016).

During this phase, it was observed that overlapping or correlated patterns had a detrimental effect on the recall rate. This was evident when multiple patterns yielded high values during a neuron update. For instance, in the scenario where 50% of the patterns exhibit neuron activity at position *j*, such noise would provide a significant challenge in retracing the original pattern from which perturbations were made.

Without any correlation between the patterns concerning the *j*-th neuron, the least achievable value is 0. This implies that if neurons are firing at position *j* in the patterns, the corresponding neurons in the input x are inactive, or no neurons are firing at location *j*.

$$\tilde{x}[j] = 0, \text{for no correlation} = \Rightarrow \tilde{x}[j] \geq 0$$

In the alternative scenario where all patterns exhibit activation in the *j*-th neuron, the resultant value can increase considerably, as it would be the total number of occurrences of the number one in each pattern. This would result in a strong correlation among all the stored concepts, as they share a common microfeature, specifically the firing of the *j*-th neuron. At the same time, the input *x* exhibits firing in all of its neurons. The purpose of analyzing this situation is simply to highlight the extensive spectrum of potential values for the updated returns; however, it is unlikely for this scenario to occur.

$$\tilde{x}[j] = n \cdot ones_i, \text{for maximum correlation} \Rightarrow \tilde{x}[j] \leq n \cdot ones_i$$

We consider $ones_i$ as the number of neurons firing in each pattern p_i.

2. Pattern Suppression

We have an input that is a perturbation of pattern no. 1; this means that once we iterate our input through the network. Because our features are closest to those of the pattern we disturbed from, pattern no. 1, we anticipate returning to the stable state it offers.

To be creative and find another pattern to connect with, we need to ensure that our network will not let our input iterate towards pattern no. 1 while at the same time keeping it part of our system (Abe, 1993). In other words, we do not erase a concept or break its connections; instead, we shift our focus to finding another solution, and this is the rationale beyond reaching the idea of suppressing a pattern.

We can suppress a pattern from our network without deleting it by subtracting its contributions from the weight matrix W, which measures the correlation between patterns as perceived by each neuron. When performing a specific neuron update, we have to reduce the value of a given pattern to disable its contribution. Therefore, the new formula for updating neuron *j* of input *x* while suppressing pattern no. k, which will be later evaluated through a threshold, is:

$$
\begin{aligned}
x[j] = x \cdot \left(\begin{bmatrix} W_{(1,j)} \\ W_{(2,j)} \\ \vdots \\ W_{(n,j)} \end{bmatrix} - p^k \cdot p_j^{(k)} \right) = \\
= (x_1, x_2, \ldots, x_N) \cdot \left(\begin{bmatrix} \sum_{i=1}^{n} p_1^{(i)} \cdot \mathbf{p_j^{(i)}} \\ \sum_{i=1}^{n} p_2^{(i)} \cdot \mathbf{p_j^{(i)}} \\ \vdots \\ \sum_{i=1}^{n} p_N^{(i)} \cdot \mathbf{p_j^{(i)}} \end{bmatrix} - p^k \cdot p_j^{(k)}) \right) = \\
= (x_1, x_2, \ldots, x_N) \cdot \begin{bmatrix} \sum_{i=1, i \neq k}^{n} p_1^{(i)} \cdot \mathbf{p_j^{(i)}} \\ \sum_{i=1, i \neq k}^{n} p_2^{(i)} \cdot \mathbf{p_j^{(i)}} \\ \vdots \\ \sum_{i=1, i \neq k}^{n} p_N^{(i)} \cdot \mathbf{p_j^{(i)}} \end{bmatrix}
\end{aligned}
$$

After this preliminary analysis, it can be inferred that the update operation yields a value that falls within, which increases significantly along with the number of patterns. As choosing

an appropriate threshold that can discretize the result of the update computation is essential for the functionality of HNN, it is imperative to understand better the actual values to make informed decisions regarding threshold selection and neuron update. Here, we refer to the **"second Knob of creativity"** to make our network reach another solution. Twisting the knob suppressed specific patterns, allowing the HNN to creatively develop a pattern to associate our input with.

Throughout the implementation process, we realized the importance of the threshold in the binary HNN. Specifically, we observed that there was only a small window of range values for the threshold within which our network could attain a stable state while simultaneously suppressing certain patterns. One possible approach to address this issue would involve utilizing the first knob that was introduced and tweaking the threshold values gradually until our network exhibits creative behavior and ultimately converges into an alternative pattern.

3. Reconstructing Associative Chains

Semantic association-based creative thinking involves connecting previously unrelated notions to develop a new path (Beaty et al., 2023; Kenett & Beaty, 2023; Li et al., 2021; Luchini et al., 2023). For example, the input is a rose, and the output is honey. How can we find their associative chain? Our primary association is with a floral entity, specifically a rose, given its striking resemblance. Upon further contemplation of the various facets of flowers, we came to pollen as our next connection. We find the correlation between bees and pollen collection by considering what else we can relate to pollen, not flowers, roses, or being trapped pollen. We have yet to attain our objective, thus we explore what bees can connect to, except for roses, flowers, and pollen, and think about honey. These connections are still considered obvious and activated by slightly different groups of neurons, known as neural cliques (Wang et al., 2003).

The situation is different once our brain's oblivious or already established connections do not solve our specific problem. Continuing the previous example, what if you have a bouquet of roses but no available vase - a new creative association could be to place them in a cup of water. When you are engaged in taking a step back from the situation and, as a result, make new connections, the group of neurons that fire together to form it are related to atypical microfeatures and are known in the neuroscience discipline as neurds (Gabora, 2010). Thus, another associative chain is created by combining one or more associative links, which tie together two concepts based on the neurons firing at common microfeatures.

Results

1. Threshold Limits

We measured the success rate by counting how many tests resulted in a perfectly reconstructed memory. This network generates “n” new patterns with 10% of randomly positioned firing neurons, perturbs a randomly selected pattern with 100 neurons, and feeds it as input.

The threshold ranges with a success rate of 100, indicating that it effectively recalls the perturbed pattern and demonstrates the decreasing size of windows as n increases in all tests (Fig. 2). For values of n greater than or equal to 8, the network was not able to achieve a complete success rate. The success rate reduces dramatically for any threshold number. Specifically, for n = 8, the maximum success rate observed was 87%. For n = 9, the success rate dropped to 72%.

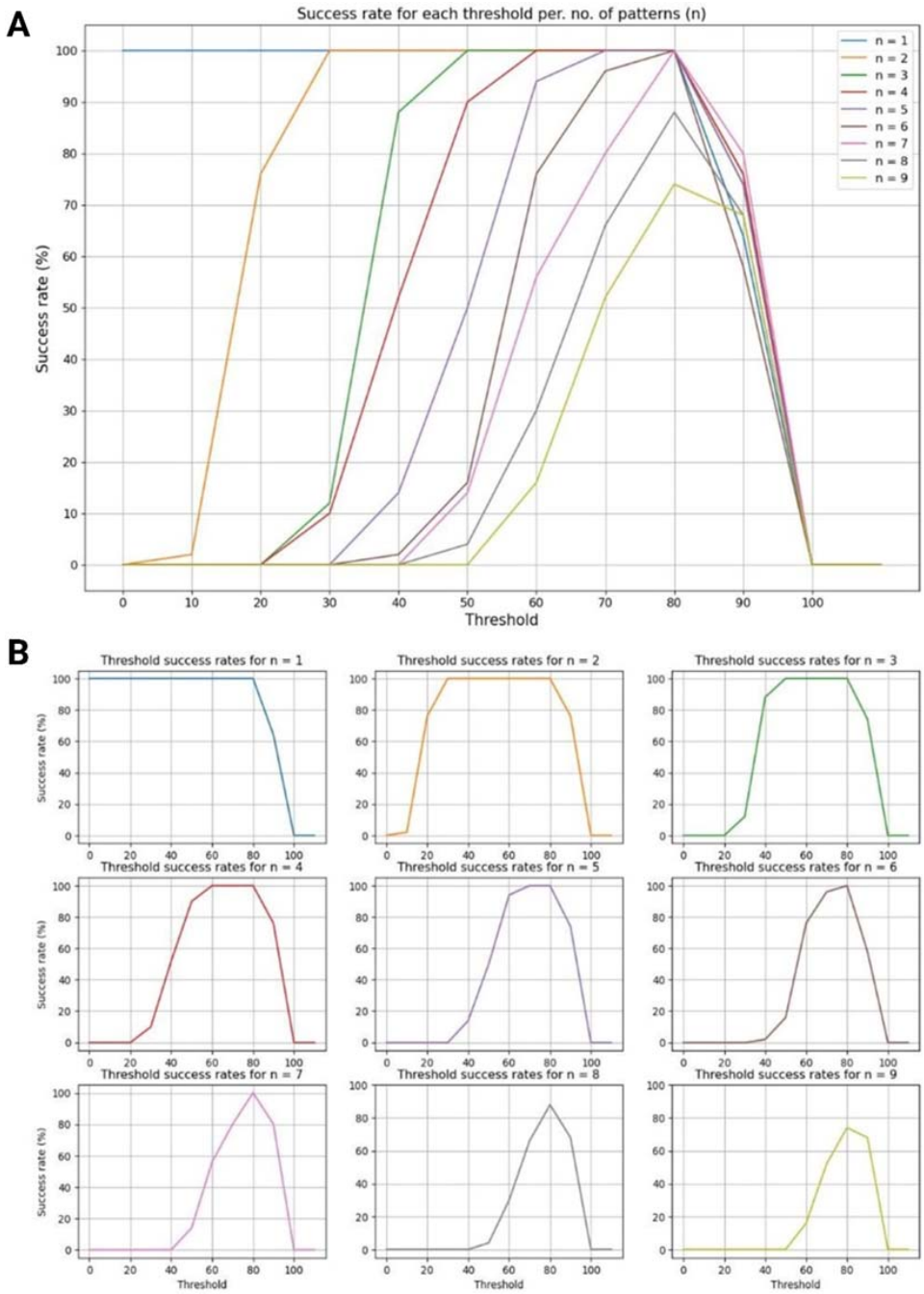

Figure 2. Illustration of measuring the success rate, quantifying the number of tests yielded in a perfectly reconstructed memory. Panel A illustrates the success rates of each threshold value for each number of patterns n. Panel B represents the success rates of each threshold value for each number of patterns n (i.e., the collected data in individual graphs, one for each value of n tested).

Remarkably, for all n for which a maximum success rate was achieved, the maximum threshold that would act is t = 80; the network completely shuts down for thresholds t ≥ 100, having a 0 recall rate. The threshold ranges of each *n*, which resulted in a perfect success rate of 100% or a good success rate (below 100 % but above 75% (Fig.3a), and the minimum and maximum values for which we achieve a complete success rate (Fig.3b).

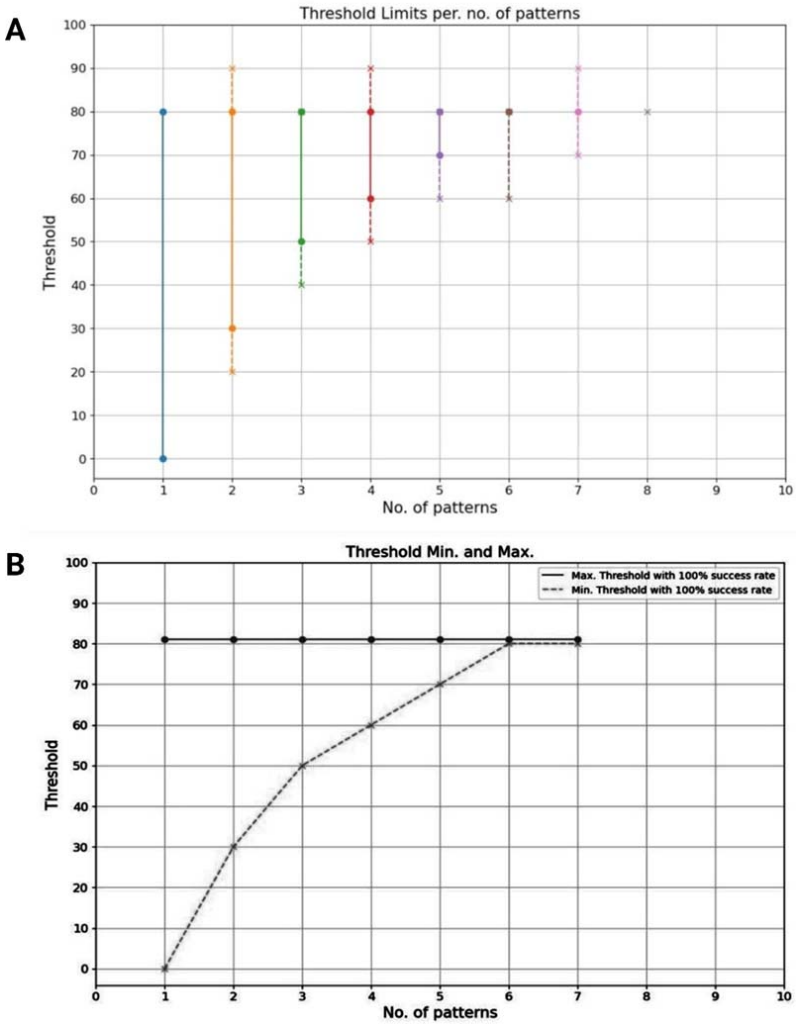

Figure 3. Depiction of the successful threshold values. Panel A illustrates threshold ranges with a perfect success rate (=100%, solid lines) and reasonable success rate (≥75% and <100%, dotted lines). Panel B displays the lowest and highest threshold values with a reasonable success rate); it visualizes the almost logarithmic decrease of

the ranges as n increases and the maximum threshold of t = 80 for a total success rate for all values of n.

This figure visualizes the almost logarithmic decrease of the ranges as n increases and the maximum threshold of $t = 80$ for a total success rate for all n.

The upper limit of the threshold can be elucidated by considering the 10% of firing neurons for each pattern and the perturbation of 100 neurons that we employ to generate our input. Thus, in each of our patterns, the probability of a neuron being 1 is 10%, while the probability of it being 0 is 90%. Therefore, if we flip 100 neurons, most likely 10% of them, which are ones, will become zeros; consequently, we have 10 neurons shutting down, and the rest of 90% of zeros will become ones; accordingly, we have 90 new firing neurons. This concludes that the input has approximately $-10 + 90 = 80$ new values of one.

When applying the thresholds, we attempt to shut down the extra neurons activated in the perturbation process to reach a stable state with approximately 80 fewer firing neurons. Therefore, it is logical to consider the maximum value obtained during the examination of the gathered data.

Finally, evaluating the gathered data underlines the network's capacity; in particular, we can have a perfect recall rate for $n \leq 7$. The deterioration of the reconstruction of patterns can be attributed to the cross-talk phenomenon, as discussed earlier. I.e., as the number of stored patterns increases, the likelihood of correlations between them also rises substantially. Consequently, retrieving a particular pattern that has been disrupted becomes more challenging.

2. First Knob of Creativity

We used the threshold as a parameter to twist this **“first knob of creativity”** to regulate whether the network can operate creatively, discover a solution, or shuts down completely.

If we twist the knob to t = 60 for n = 3 memorized patterns, the network will act creatively and find a solution, but if we twist it to t = 35, it will shut off and not provide a solution. The network will act creatively for n = 5 patterns and find a solution if we twist it to t = 70, Fig.4.

```
No. of differing neurons between the updated input and pattern  0  is:  0
No. of differing neurons between the updated input and pattern  1  is:  182
No. of differing neurons between the updated input and pattern  2  is:  188
Starting to twist the knob...

Testing threshold:  15
No. of differing neurons between the updated input and pattern  0  is:  256
No. of differing neurons between the updated input and pattern  1  is:  88
No. of differing neurons between the updated input and pattern  2  is:  88
--- Smallest no. of differing neurons is:  88
Testing threshold:  15.5
No. of differing neurons between the updated input and pattern  0  is:  256
No. of differing neurons between the updated input and pattern  1  is:  88
No. of differing neurons between the updated input and pattern  2  is:  88
--- Smallest no. of differing neurons is:  88
Testing threshold:  16.0
No. of differing neurons between the updated input and pattern  0  is:  256
No. of differing neurons between the updated input and pattern  1  is:  88
No. of differing neurons between the updated input and pattern  2  is:  88
--- Smallest no. of differing neurons is:  88
Testing threshold:  16.5
No. of differing neurons between the updated input and pattern  0  is:  182
No. of differing neurons between the updated input and pattern  1  is:  0
No. of differing neurons between the updated input and pattern  2  is:  176
--- Smallest no. of differing neurons is:  0
Threshold is:  16.5
```

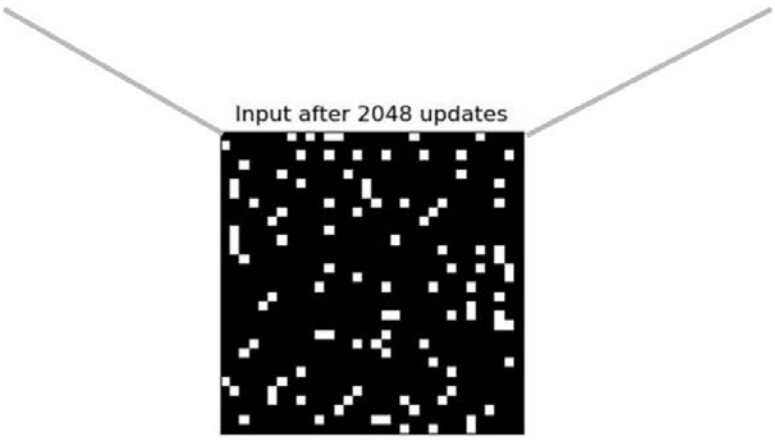

Figure 4. Illustratation if an instance where pattern suppression is effectively achieved. Initially, the input iterates towards pattern number 0, but upon suppressing this pattern by adjusting the first knob of creativity controlling the threshold, the input subsequently iterates towards pattern number 1.

3. The Second Knob of Creativity

We feed input into our network, which iterates toward its most associated pattern (Fig.5), in our case, pattern no. 6, establishing our first associative connection.

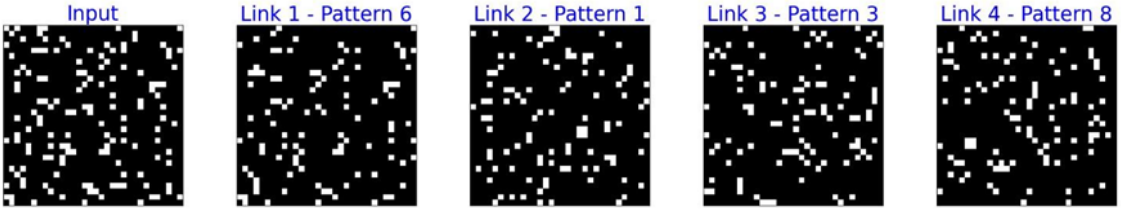

Figure 5. Depiction of the input pattern exhibits the highest correlation, indicating distinct associative chains represented in various links. Pattern 6 is responsible for generating the initial associative link. We will input pattern number 6 into our network to locate the desired link while simultaneously inhibiting it. As a result, HNN is compelled to establish an association between pattern number 6 and another. Consequently, the generation of the second link with pattern number 1 occurs. The validity of this new link is based on the assumption that there is a correlation between pattern number 6 and the initial input. Accordingly, the fundamental nature of the input is maintained during the process of identifying the future relationship. To extend the associative chain, we include pattern no. 1 as an input while excluding itself and pattern no. 6. This approach guarantees the identification of subsequent links, specifically with pattern no. 3, and so on. For a better understanding of these associations, see Figure 7.

We will suppress pattern 6 and feed it to our network to force the HNN to find a relationship between it and another pattern. The second link, with pattern no. 1, is generated. This new link is valid because pattern no. 6 and the initial input have correlated features; therefore, the input kernel is not lost when finding the next association. For a longer associative chain, supply pattern 1 as an input while suppressing itself and pattern 6, to locate the next link, especially with pattern 3, etc.

To advance our research, we evaluate our conceptual framework further using more patterns. However, the limited capacity of the binary HNN impeded our progress, prompting us to delve further into the modern HNN.

4. Modern Hopfield Network

The modern HNN was developed a few years ago, aiming to support continuous states implementing a new update rule and its respective energy function (Koiran, 1994; Krotov & Hopfield, 2016; Ramsauer et al., 2021; Šíma et al., 2000). Therefore, we introduced our perspective by rearranging the HNN into a two-layer network (Fig.6).

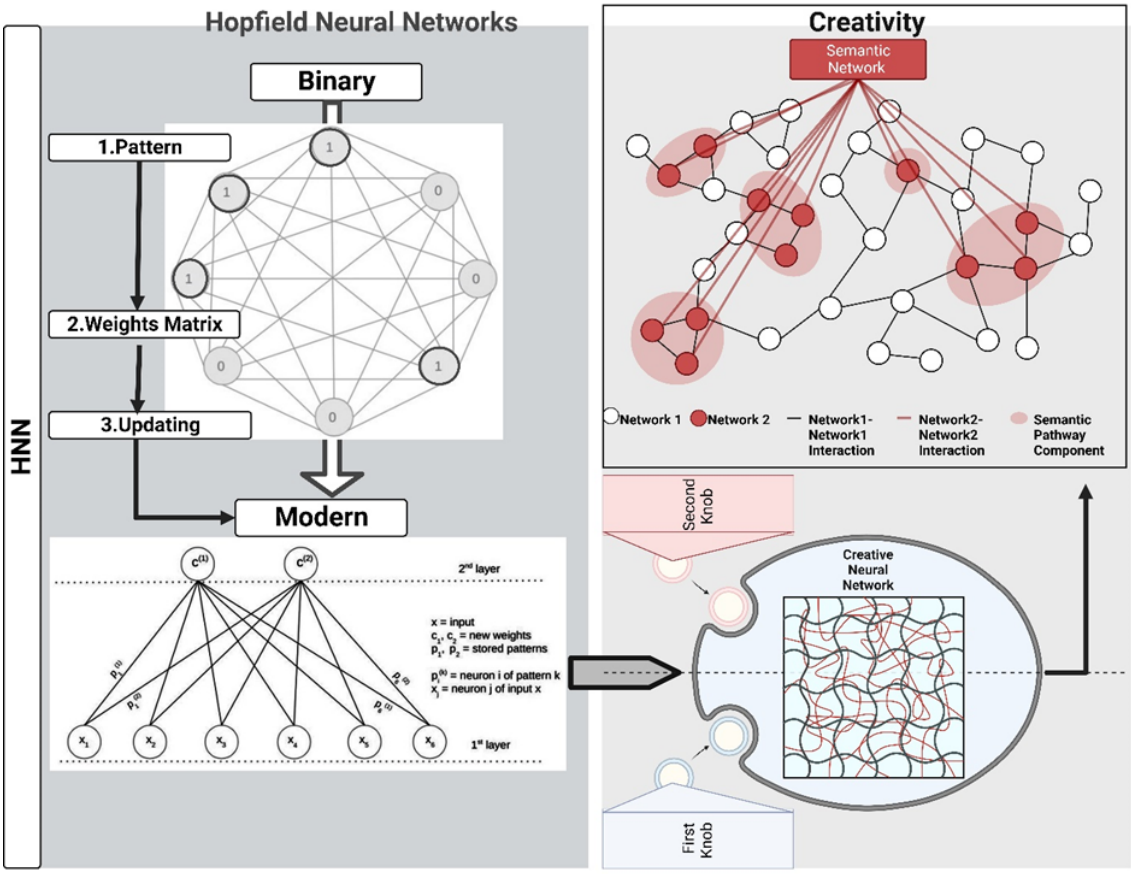

Figure 6: Summary of a hypothetical model of the experiment. Initially, we started with binary HNN, and as a result, we decided on modern HNN as a suitable model to represent the associative chains of creativity. We created modern HNN as a two-layer model. The first layer comprises the neurons of the input, and the connections between the two layers are realized through the neurons of the patterns. The top layer contains the core part of the updating process, which may be regarded as the "new weights," denoted as c(i). The 'new weights' are a computation of the correlation between the current state of the input x and each pattern, later put through a normalization

performed by the softmax function. Accordingly, this resembles the first and second Knobs of creativity, symbolized as a semantic association.

The first layer consists of input neurons, whereas the interconnections between the two layers are established through the neurons representing patterns. The second layer encompasses the fundamental aspect of the updating process, which we may presently regard as the "new weights," denoted as c(i). These "new weights" are derived by calculating the correlation between the current state of the input variable x and each pattern. These weights are then normalized using the softmax function.

$$c^{\sim(j)} = x \cdot p^{(j)}$$

$$c^{(j)} = \mathrm{softmax}(c^{\sim(j)})$$

The softmax normalization function is defined as follows:

$$softmax\ (c^{\sim x(j)}) = \frac{e^{c^{(j)}}}{\sum_(i\ =\ 1)\text{^}n\ e\text{^}(c\text{^}((i))\ )\ )}$$

At the start of each asynchronous update, c was calculated to discover the most correlated pattern. The softmax function normalizes the pattern we should iterate; thus, setting an equal threshold is unnecessary. Accordingly, all current HNN implementations used t = 0.5.

We achieved a significantly higher recall rate and increased storage capacity using the modern HNN. We tested our network and got a perfect success rate even when storing 100 patterns. This first step is impressive compared to the simple HNN with 8 stored patterns as a capacity. The new updating rule improves performance using patterns' neurons instead of the weights matrix. In modern HNNs, 'new weights' function as neurons, storing a single value from 0 to 1 due to normalization and dictating the idea activated, similar to a 'decision' neuron.

After successfully implementing the two HNN layers, we focused on associative chains. The most crucial part of their creation was suppressing a pattern in this new framework. We shut off ci = 0, the "decision" neuron in control of the thought we want to suppress, similar to

memory reconstruction. Thus, we retained its linkages but focused on the other concepts, creating associative chains linking previously unrelated concepts in a framework with a higher capacity.

Discussion and Concluding Remarks

We created associative connections between previously unconnected concepts using HNN to create a creativity-based semantic network conceptual framework (Fig.6). We started with simple binary and bipolar HNN and found the significance of the threshold; after performing a data analysis collection by testing several threshold values for different numbers of patterns, we defined the **"first knob of creativity"**, which controls whether the network acts creatively (i.e., finds a new path) or completely shuts down.

We also uncovered the capacity of the network n = 8, namely how many patterns we could store while still having a perfect recall rate. Here, we questioned what more we could do to help our system act creatively and reach another solution. At this stage, we defined suppressing a pattern such that it still exists in our binary HNN model and makes its connections. We stepped the theoretical step back and focused on finding another connection; thus, we established the **"second knob of creativity"** to assist our network in finding a solution other than suppression. Being constrained by the small capacity of the binary HNN, we transitioned to a modern HNN consisting of two layers. After redefining the suppression of a pattern in the new context, we successfully generated associations and tested our implementation using more patterns.

The first and second knobs assisted us in creating associative links between previously unrelated concepts and further developing them computationally into associative chains connecting several concepts; for a better understanding of the concept, see Fig.7.

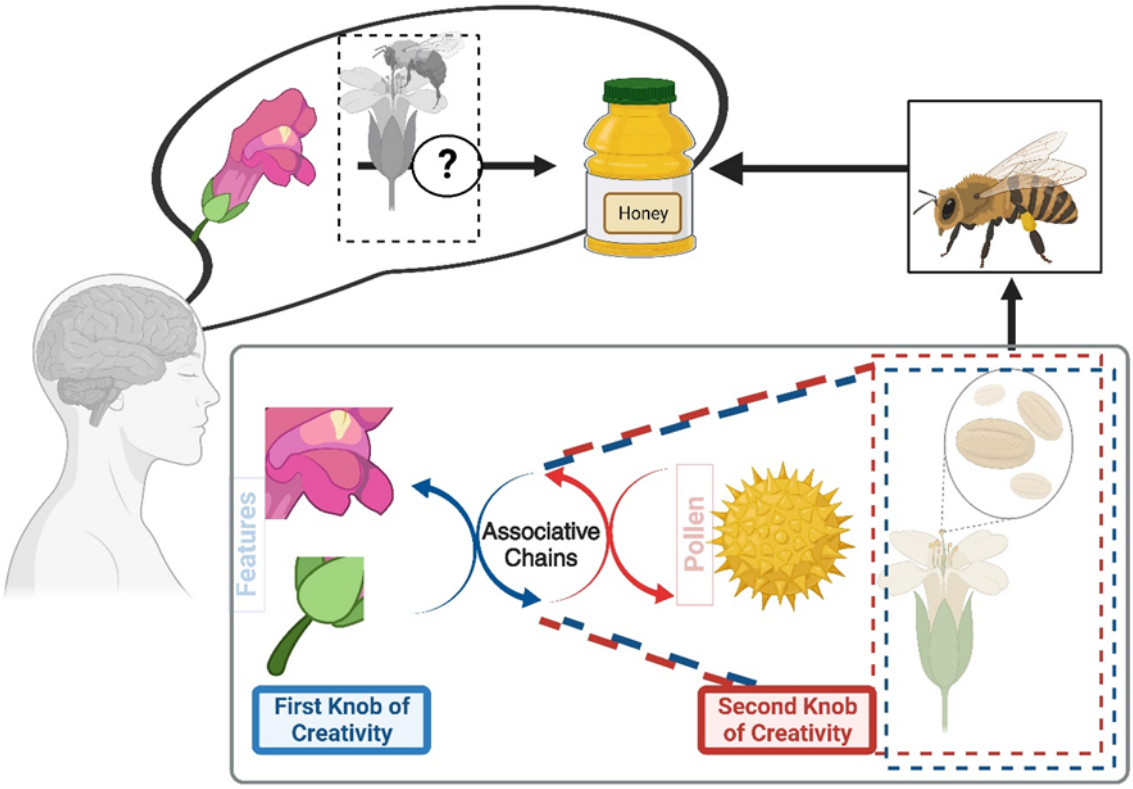

Figure 7. An illustrative case demonstrating the link of concepts and the potential extent of our associative chain without imposing a preconceived restriction on these links. If we were to examine the input "a rose" and the desired endpoint "honey," what would be the associative chains? Which strategies can be utilized to determine the connection between these entities? The rose is classified as a flower and is mainly associated with the concept of flowers. Therefore, our initial connection is established with the category of flowers. In this context, the focus is directed towards the attributes of a flower, which may be considered the initial catalyst for **the first knob of creativity**. Pollen emerges as the subsequent element, representing the second facet of creative exploration (i.e., **the second knob of creativity**). In this context, we construct associative chains and engage in cognitive processes to identify additional associations with pollen while avoiding simplistic connections such as flowers, roses, or the pollen itself. Consequently, we discover a correlation between bees and their role in pollen collection. The desired objective has yet to be achieved, so further exploration is undertaken to ascertain the factors bees associate with accessing honey, including the bee, flower, and pollen.

In conclusion, we successfully presented for the first time a neurocomputational framework model for creativity-based semantic associations, being implemented in both a

simple, binary HNN and a modern, two-layer one (Fig.6 and Fig.7). A potential explanation of our findings is that our neurons take a step back from the contextual focus and find alternatives associated with seemingly unrelated concepts (Beaty & Kenett, 2023; Gabora, 2010; Krotov & Hopfield, 2016; McEliece et al., 1987; Mednick, 1962) when analytical thinking is insufficient.

Accordingly, we do not have to search the whole network and compare our input with all the patterns to find a solution; it helps us retrieve the memorized items seamlessly. Therefore, an associative link is created when a group of neurons (i.e., neural cliques; Wang et al., 2003) fire up together while being part of both concepts we are connecting; this allows us to find a solution that would not be an oblivious choice but that solves our problem This could be analog of creative thinking based on semantic associations, allowing us to search on the lower levels of memories to find which concepts relate even on a micro-level to our situation (Beaty et al., 2023; Kenett & Beaty, 2023; Li et al., 2021; Luchini et al., 2023). The patterns and how we store and reconstruct them align with the biological processes, as human memory is also distributed, coarse-coded, and content-addressable (Beaty et al., 2023; Benedek et al., 2023; Gerver et al., 2023; McEliece et al., 1987).

This preliminary computational neural network model has numerous possibilities for investigation, as we can find creative ways to connect concepts and see how far our chain can reach without imposing a predetermined limit on the extent of these connections.

Future Direction

Several interesting open questions need to be addressed in the upcoming research work; here is one of them. It would be ideal for testing the framework on patterns representative of real-life concepts (i.e., create a dataset for them and define a set of multiple features that should be defined for each of them (e.g., human/animal/object, weight, height, color, etc.)) as

a next step, daily creativity (Richards, 1993; Runco & Bahleda, 1986); however, it is challenging empirically. However, it is possible through implementing an algorithm that takes as input a concept, and after the user answers questions about its features, we can collect the data and transform it into a pattern. We understand the idea of a feature as a group of neurons (e.g., the first 10 neurons of each pattern) and the definition of it as the positions of the firing neurons (e.g., a neuron firing on the second position in a weight feature translates to a light object, while 0 means a weight is not applicable).

Therefore, a natural and crucial next step would be testing this theoretical framework on concepts modeled by real-life concepts; accordingly, the network's creativity-based semantic associations will also be easier to observe. Taking it a step further, we could also redefine the way associative links get created by iterating solely on the defined features and, as such, suppressing certain features.

References:

Abdullah, M., Madain, A., & Jararweh, Y. (2022). ChatGPT: Fundamentals, Applications and Social Impacts. *2022 Ninth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, 1–8. https://doi.org/10.1109/SNAMS58071.2022.10062688

Abe. (1989). Theories on the Hopfield neural networks. *International Joint Conference on Neural Networks*, p. 557–564 vol.1. https://doi.org/10.1109/IJCNN.1989.118633

Abe, S. (1993). Global convergence and suppression of spurious states of the Hopfield neural networks. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, *40*(4), 246–257. https://doi.org/10.1109/81.224297

Abe, S., & Gee, A. H. (1995). Global convergence of the Hopfield neural network with nonzero diagonal elements. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, *42*(1), 39–45. https://doi.org/10.1109/82.363543

Abraham, A. (2013). The promises and perils of the neuroscience of creativity. *Frontiers in Human Neuroscience*, *7*. https://doi.org/10.3389/fnhum.2013.00246

Abraham, A. (2018). *The Neuroscience of Creativity*. Cambridge University Press. https://doi.org/10.1017/9781316816981

Amari, S.-I., & Maginu, K. (1988). Statistical neurodynamics of associative memory. *Neural Networks*, *1*(1), 63–73. https://doi.org/10.1016/0893-6080(88)90022-6

Amit, D. J. (1995). The Hebbian paradigm reintegrated: Local reverberations as internal representations. *Behavioral and Brain Sciences*, *18*(4), 617–626.

https://doi.org/10.1017/S0140525X00040164

Anishchenko, A., & Treves, A. (2006). Autoassociative memory retrieval and spontaneous activity bumps in small-world networks of integrate-and-fire neurons. *Journal of Physiology-Paris*, *100*(4), 225–236. https://doi.org/10.1016/j.jphysparis.2007.01.004

Atencia, M., Joya, G., & Sandoval, F. (2005). Hopfield Neural Networks for Parametric Identification of Dynamical Systems. *Neural Processing Letters*, *21*(2), 143–152. https://doi.org/10.1007/s11063-004-3424-3

Aviel, Y., Horn, D., & Abeles, M. (2004). The doubly balanced network of spiking neurons: A memory model with high capacity. *Advances in Neural Information Processing Systems*.

Beaty, R. E., & Kenett, Y. N. (2023). Associative thinking at the core of creativity. *Trends in Cognitive Sciences*, *27*(7), 671–683. https://doi.org/10.1016/j.tics.2023.04.004

Beaty, R. E., Kenett, Y. N., Hass, R. W., & Schacter, D. L. (2023). Semantic memory and creativity: the costs and benefits of semantic memory structure in generating original ideas. *Thinking & Reasoning*, *29*(2), 305–339. https://doi.org/10.1080/13546783.2022.2076742

Benedek, M., Beaty, R. E., Schacter, D. L., & Kenett, Y. N. (2023). The role of memory in creative ideation. *Nature Reviews Psychology*, *2*(4), 246–257. https://doi.org/10.1038/s44159-023-00158-z

Brunel, N. (1996). Hebbian Learning of Context in Recurrent Neural Networks. *Neural Computation*, *8*(8), 1677–1710. https://doi.org/10.1162/neco.1996.8.8.1677

Corazza, G. E. (2016). Potential Originality and Effectiveness: The Dynamic Definition of Creativity. *Creativity Research Journal*, *28*(3), 258–267. https://doi.org/10.1080/10400419.2016.1195627

Davison, A. P. (2008). PyNN: a common interface for neuronal network simulators. *Frontiers in Neuroinformatics*, *2*. https://doi.org/10.3389/neuro.11.011.2008

Gabora, L. (2010). Revenge of the "Neurds": Characterizing Creative Thought in Terms of the Structure and Dynamics of Memory. *Creativity Research Journal*, *22*(1), 1–13. https://doi.org/10.1080/10400410903579494

Gardner. (1994). *Creating Minds: An Anatomy of Creativity as Seen Through the Lives of Freud, Einstein, Picasso, Stravinsky, Eliot, Graham, and Gandhi*.

Gaut, B. (2010). The Philosophy of Creativity. *Philosophy Compass*, *5*(12), 1034–1046. https://doi.org/10.1111/j.1747-9991.2010.00351.x

Gerver, C. R., Griffin, J. W., Dennis, N. A., & Beaty, R. E. (2023). Memory and creativity: A meta-analytic examination of the relationship between memory systems and creative cognition. *Psychonomic Bulletin & Review*. https://doi.org/10.3758/s13423-023-02303-4

Han, J., Forbes, H., & Schaefer, D. (2021). An exploration of how creativity, functionality, and aesthetics are related in design. *Research in Engineering Design*, *32*(3), 289–307. https://doi.org/10.1007/s00163-021-00366-9

Herrmann, N. (1981). The Creative Brain. *Training and Development* , *35*(10), 10–16.

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

Joya, G., Atencia, M. A., & Sandoval, F. (2002). Hopfield neural networks for optimization: study of the different dynamics. *Neurocomputing*, *43*(1–4), 219–237. https://doi.org/10.1016/S0925-2312(01)00337-X

Kappas, A., & Gratch, J. (2023). These Aren ' t The Droids You Are Looking for?: Promises and Challenges for the Intersection of Affective Science and Robotics / AI. *Affective Science*, *0123456789*. https://doi.org/10.1007/s42761-023-00211-3

Kaufman, J. C., & Beghetto, R. a. (2009). Beyond Big and Little: The Four C Model of Creativity. *Review of General Psychology*, *13*(1), 1–12. https://doi.org/10.1037/a0013688

Kenett, Y. N., & Beaty, R. E. (2023). On semantic structures and processes in creative thinking. *Trends in Cognitive Sciences*. https://doi.org/10.1016/j.tics.2023.07.011

Khalil, R., & Demarin, V. (2023). Creative therapy in health and disease: Inner vision. *CNS Neuroscience & Therapeutics*. https://doi.org/10.1111/cns.14266

Khalil, R., & Moustafa, A. A. (2022). A neurocomputational model of creative processes. *Neuroscience & Biobehavioral Reviews*, *137*, 104656. https://doi.org/10.1016/J.NEUBIOREV.2022.104656

Kobayashi, M. (2021). Complex-valued Hopfield neural networks with real weights in synchronous mode. *Neurocomputing*, *423*, 535–540. https://doi.org/10.1016/j.neucom.2020.10.072

Koiran, P. (1994). Dynamics of Discrete Time, Continuous State Hopfield Networks. *Neural Computation*, *6*(3), 459–468. https://doi.org/10.1162/neco.1994.6.3.459

KÖksal, S., & Sivasundaram, S. (1993). Stability properties of the Hopfield-type neural networks. *Dynamics and Stability of Systems*, *8*(3), 181–187. https://doi.org/10.1080/02681119308806157

Krotov, D., & Hopfield, J. J. (2016). Dense associative memory for pattern recognition. *Advances in Neural Information Processing Systems*, *Nips*, 1180–1188.

Li, Y., Kenett, Y. N., Hu, W., & Beaty, R. E. (2021). Flexible Semantic Network Structure Supports the Production of Creative Metaphor. *Creativity Research Journal*, *33*(3), 209–223. https://doi.org/10.1080/10400419.2021.1879508

Litinskii, L. B. (2001). Hopfield Model with a Dynamic Threshold. *Theoretical and Mathematical Physics*, *130*(1), 136–151. https://doi.org/10.1023/A:1013840801300

Luchini, S., Kenett, Y. N., Zeitlen, D. C., Christensen, A. P., Ellis, D. M., Brewer, G. A., & Beaty, R. E. (2023). Convergent thinking and insight problem solving relate to semantic memory network structure. *Thinking Skills and Creativity*, *48*, 101277. https://doi.org/10.1016/j.tsc.2023.101277

Mastria, S., Agnoli, S., Corazza, G. E., Grassi, M., & Franchin, L. (2022). What inspires us? An experimental analysis of the semantic meaning of irrelevant information in creative ideation. *Thinking and Reasoning*, *0*(0), 1–28. https://doi.org/10.1080/13546783.2022.2132289

McEliece, R., Posner, E., Rodemich, E., & Venkatesh, S. (1987). The capacity of the Hopfield associative memory. *IEEE Transactions on Information Theory*, *33*(4), 461–482. https://doi.org/10.1109/TIT.1987.1057328

Mednick, S. (1962). The associative basis of the creative process. *Psychological Review*, *69*(3), 220–232. https://doi.org/10.1037/h0048850

Mendelsohn, G. A. (1976). Associative and attentional processes in creative performance. *Journal of*

Personality, *44*(2), 341–369. https://doi.org/10.1111/j.1467-6494.1976.tb00127.x

Michel, A. N., Farrell, J. A., & Sun, H.-F. (1990). Analysis and synthesis techniques for Hopfield-type synchronous discrete time neural networks with application to associative memory. *IEEE Transactions on Circuits and Systems*, *37*(11), 1356–1366. https://doi.org/10.1109/31.62410

Morris, R. G. . (1999). D.O. Hebb: The Organization of Behavior, Wiley: New York; 1949. *Brain Research Bulletin*, *50*(5–6), 437. https://doi.org/10.1016/S0361-9230(99)00182-3

Ramsauer, H., Schäfl, B., Lehner, J., Seidl, P., Widrich, M., Adler, T., Gruber, L., Holzleitner, M., Kreil, D., Kopp, M., Klambauer, G., Brandstetter, J., & Hochreiter, S. (2021). Hopfield Networks Is All You Need. *ICLR 2021 - 9th International Conference on Learning Representations*.

Richards, R. (1993). Everyday Creativity, Eminent Creativity, and Psychopathology. *Psychological Inquiry*, *4*(3), 212–217. https://doi.org/10.1207/s15327965pli0403_12

Runco, M. A., & Bahleda, M. D. (1986). Implicit Theories of Artistic, Scientific, and Everyday Creativity*. *The Journal of Creative Behavior*, *20*(2), 93–98. https://doi.org/10.1002/j.2162-6057.1986.tb00423.x

Runco, M. A., & Jaeger, G. J. (2012). The Standard Definition of Creativity. *Creativity Research Journal*, *24*(1), 92–96. https://doi.org/10.1080/10400419.2012.650092

Sawyer. (2006). Explaining Creativity - The Science if Human Innovation. In *Creativity and Consciousness: Philosophical and ….* https://doi.org/10.1016/0140-1750(88)90050-4

Sawyer. (2011). The Cognitive Neuroscience of Creativity: A Critical Review. *Creativity Research Journal*, *23*(2), 137–154. https://doi.org/10.1080/10400419.2011.571191

Shneiderman, B. (2020). Design Lessons From AI's Two Grand Goals: Human Emulation and Useful Applications. *IEEE Transactions on Technology and Society*, *1*(2), 73–82. https://doi.org/10.1109/TTS.2020.2992669

Šíma, J., Orponen, P., & Antti-Poika, T. (2000). On the Computational Complexity of Binary and Analog Symmetric Hopfield Nets. *Neural Computation*, *12*(12), 2965–2989. https://doi.org/10.1162/089976600300014791

Squalli, J., & Wilson, K. (2014). Intelligence, creativity, and innovation. *Intelligence*, *46*(1). https://doi.org/10.1016/j.intell.2014.07.005

Tardif, T. Z., & Sternberg, R. J. (1988). What do we know about creativity? In *The Nature of Creativity: Contemporary Psychological Perspectives*.

Torrance, E. P. (1974). The Torrance Tests of Creative Thinking: Norms-Technical Manual. Research Edition. Verbal Tests, Forms A and B. Figural Tests, Forms A and B. Princeton, NJ: Personnel Press. *Research Edition*.

Virvou, M. (2022). The Emerging Era of Human-AI Interaction: Keynote Address. *2022 13th International Conference on Information, Intelligence, Systems & Applications (IISA)*, 1–10. https://doi.org/10.1109/IISA56318.2022.9904422

Wang, D., Feng, L., Ye, J., Zou, J., & Zheng, Y. (2023). Accelerating the integration of ChatGPT and other large-scale AI models into biomedical research and healthcare. *MedComm – Future Medicine*, *2*(2). https://doi.org/10.1002/mef2.43

Wang, I-Hsum Li, Wei-Ming Wang, Shun-Feng Su, & Nai-Jian Wang. (2005). A New Convergence Condition for Discrete-Time Nonlinear System Identification Using a Hopfield Neural Network.

IEEE International Conference on Systems, Man and Cybernetics, *1*, 685–689. https://doi.org/10.1109/ICSMC.2005.1571226

Wang, Tang, Z., & Cao, Q. P. (2003). An Efficient Approximation Algorithm for Finding a Maximum Clique Using Hopfield Network Learning. *Neural Computation*, *15*(7), 1605–1619. https://doi.org/10.1162/089976603321891828

Weisbuch, G., & Fogelman-Soulie, F. (1985). Scaling Laws for the Attractors of Hopfield Networks. *Journal de Physique. Lettres*, *46*(14), 623–630. https://doi.org/10.1051/jphyslet:019850046014062300

Wilson, R. C., Guilford, J. P., Christensen, P. R., & Lewis, D. J. (1954). A factor-analytic study of creative-thinking abilities. *Psychometrika*, *19*(4), 297–311. https://doi.org/10.1007/BF02289230

Zhang, X., Mao, X., Yin, Y., Chai, C., & Zhang, T. (2022). Melting Your Models: An Integrated AI-based Creativity Support Tool for Inspiration Evolution. *2022 15th International Symposium on Computational Intelligence and Design (ISCID)*, 97–101. https://doi.org/10.1109/ISCID56505.2022.00029