To appear in Neural Computation, Volume 31, Issue 5, 2019

Sparse Associative Memory

Heiko Hoffmann[1]

[1]HRL Laboratories, LLC, 3011 Malibu Canyon Rd, Malibu, CA 90265

Keywords: Associative memory, sparse encoding, Hopfield network

Abstract

It is still unknown how associative biological memories operate. Hopfield networks are popular models of associative memory, but suffer from spurious memories and low efficiency. Here, we present a new model of an associative memory that overcomes these deficiencies. We call this model Sparse Associative Memory (SAM) because it is based on sparse projections from neural patterns to pattern-specific neurons. These sparse projections have shown to be sufficient to uniquely encode a neural pattern. Based on this principle, we investigate theoretically and in simulation our SAM model, which turns out to have high memory efficiency and a vanishingly small probability of spurious memories. This model may serve as a basic building block of brain functions involving associative memory.

1 Introduction

Associative memory is one of the main brain functions. The operation of biological associative memories is still unknown, but evidence suggests that associative memory is governed by attractor dynamics (Wills et al., 2005). Networks with attractor dynamics iterate a memory state until observing a stable pattern that matches a stored one. The most influential computational model of such networks has been the Hopfield network (Hopfield, 1982).

A Hopfield network is a fully-connected recurrent neural network with weighted connections between neurons, governed usually by discrete-time synchronous dynamics. Typically, the weights are symmetric, and the resulting energy function (Hamiltonian) guides the neural activation to the network's stable patterns. Hopfield networks have demonstrated a key feature of associative memories, the completion of partial patterns and reconstruction of perturbed patterns. Unfortunately, however, this network model has three major drawbacks: first, being fully connected is biologically unrealistic (Minai & Levy, 1993; Lansner, 2009). Moreover, experiments suggest that sparsely distributed activity patterns encode information (Barth & Poulet, 2012), whereas in the Hopfield network, a large fraction of neurons is active. Second, the efficiency of storage is low, i.e., the number of patterns that can be stored is low relative to the number of information needed (McEliece et al., 1987), and, third, the network suffers from spurious memories, i.e., with probability close to 1, a random input results in a stable neural activity that differs from any previously learned pattern (Bruck & Roychowdhury, 1990).

There have been many efforts to improve the efficiency of Hopfield networks (e.g., by Storkey (1997) and Hopfield (2008)), but they still suffer from spurious memories

when presented with random input. A recent variant of Hopfield networks is dense associative memory (Krotov & Hopfield, 2016), where the capacity of the network can be improved by using higher-order polynoms for the energy function. The theory, however, assumes equal probability for active versus non-active neurons, and the network performs poorly if only a small percentage of input neurons is active, which is at odds with the sparse activations found in the brain (Barth & Poulet, 2012). Another line of research looked at different architectures for associative memory, e.g., organizing the memory network into cliques, dividing neurons into clusters, which improved efficiency (Gripon & Berrou, 2011; Mofrad & Parker, 2017), but restricts learning to patterns activating one neuron per cluster. Another example is using the Hopfield dynamics on sparse networks, e.g., scale-free networks, which can increase capacity but at the cost of higher retrieval errors (Kim et al., 2017).

To create a computational model of associative memory that overcomes the above problems, we start with the concept that a single neuron can reliably detect a neural pattern even with only relatively few synaptic connections (Hawkins & Ahmad, 2016). Hawkins & Ahmad (2016) demonstrate that with high probability two neural patterns can be distinguished from each other based solely on the activation of a small subsample of active neurons. Here, we investigate this concept for creating associative memories. We demonstrate a new sparse associative memory that has much better memory efficiency and a greatly reduced probability of spurious activations. When adding inhibition to the recurrent dynamics, we found that the recurrent dynamics eliminate spurious activations in each iteration step until convergence.

The remainder of this article is organized as follows. Section 2 describes the new

sparse associative memory (SAM) and presents two variants, one with and one without inhibition. Section 3 shows the three main results: first, the theoretical computation of the memory's retrieval performance and dependence on model parameters, second, an analysis of the memory efficiency, and, third, simulation results comparing SAM against the Hopfield network and showing the benefit with respect to spurious memories. Finally, Section 4 concludes with a discussion of related work and highlights key results.

2 Sparse Associative Memory

Our sparse associative memory consists of an input layer of n neurons, a layer of hidden neurons, and connections between the two layers. The memory operates in two modes: training and recall.

Initially, before any training, the memory has an input layer of n neurons without hidden neurons or connections. In training, patterns are stored one at a time. A binary pattern to be stored activates a subset of the input layer neurons. To store the pattern, a set of h hidden neurons is created. Then, connections are formed between the input and hidden neurons (Figure 1). Herein lies the main difference to Hopfield or auto-associative memory networks (Hassoun, 1993), which map from the input neuron onto the same or an equivalent set of n neurons. Instead, we sparsely project onto pattern-specific neurons (Hawkins & Ahmad, 2016).

In our model, connections between input layer and hidden layer are formed probabilistically. With a given probability, p_S, a directed connection is formed from an active

neuron in the input layer to a hidden neuron. This formation of connections is a form of Hebbian learning - neurons wire together if they fire together (Löwel & Singer, 1992). Forming connections probabilistically differs from the model by Hawkins & Ahmad (2016), who instead use a given number of connections. Using a given number is biologically less plausible because it would require a computational mechanism to count the number of connections during formation.

After connecting the hidden neurons, we create a second set of connections that project from the hidden neurons back onto the input neurons (Figure 1): each hidden neuron assigned to the stored pattern projects to each active neuron. For the next training pattern, a new set of h hidden neurons is created, and connections are formed as described above. Therefore, in total, we have Nh hidden neurons, where N is the number of stored patterns.

We consider two settings of our memory, one without inhibitory recurrent connections and one with. For the first, the projections from the hidden neurons are only excitatory and connect only to the active neurons (Figure 1a). For the latter, additional inhibitory connections are formed between the hidden neurons and the inactive input neurons (Figure 1b). Here, a hidden neuron may project to all inactive input neurons or only to a subset of these neurons. For simplicity, we investigated only the case where the inhibitory connections project to all inactive input neurons.

In recall, a pattern is presented to the input neurons, and the associative memory iterates the activations of the input and hidden neurons until a stable pattern emerges. The neural dynamics are modeled as McCulloch-Pitts neurons: if a neuron fires, it sends a spike with value +1 through all its outgoing connections (Hoffmann et al., 2011). An

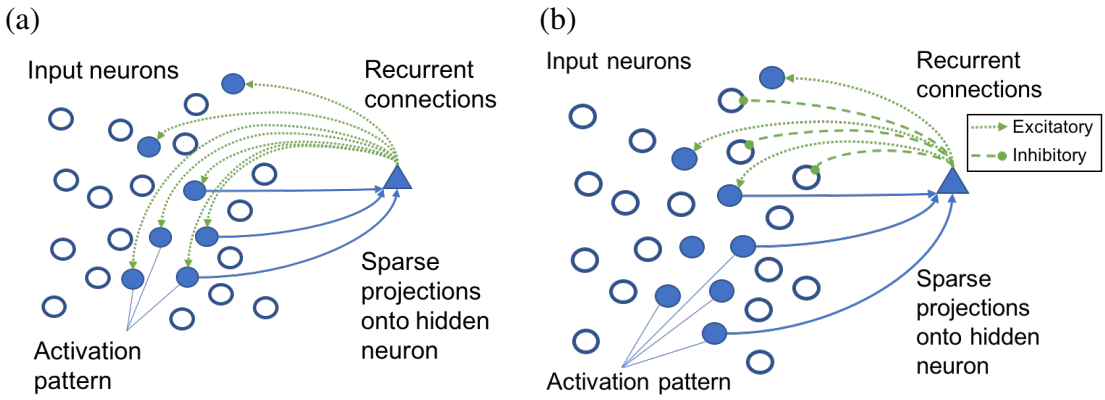

Figure 1: Associative memory with only excitatory (a) or excitatory and inhibitory (b) connections between input and hidden neurons. For clarity, in (b), not all recurrent connections are shown.

inhibitory connection multiplies this value by -1. At the receiving neuron, all incoming spike values are added together; if the sum is above a predetermined threshold, then the receiving neuron becomes active and fires. For the hidden neurons, we use a variable threshold of θ and for the input neurons a threshold of 1.

Let $\{x_i\}$ be a new input pattern for recall, where x_i is the activity of neuron i. Moreover, let W_{ji} be the binary connectivity matrix between input and hidden neurons. Then, the activity, $y_j(t)$ of a hidden neuron, j, at iteration step t, is computed as

$$y_j(t) = H\left(\sum_{i=1}^{n} W_{ji}x_i(t) - \theta\right) \quad \forall j\,, \tag{1}$$

where H is the Heaviside step function with $H(x \geq 0) = 1$. The hidden neurons then activate again the input neurons through the connectivity matrix U_{kj}, where $U_{kj} = -1$ for inhibitory connections,

$$x_k(t+1) = H\left(\sum_{j=1}^{Nh} U_{kj}y_j(t) - 1\right) \quad \forall k\,. \tag{2}$$

We iterate $x(t+1) = f(x(t))$, with f defined through Eqs. (1) and (2), for a given

number of steps or until convergence, i.e., $f(x(t)) = x(t)$.

Figure 2 shows an example of storing three patterns into our sparse associative memory with inhibition. Here, two hidden neurons were used per pattern. After storage, noisy versions of the training patterns were presented as input, and the network restored these patterns to their original shape, using a single iteration step.

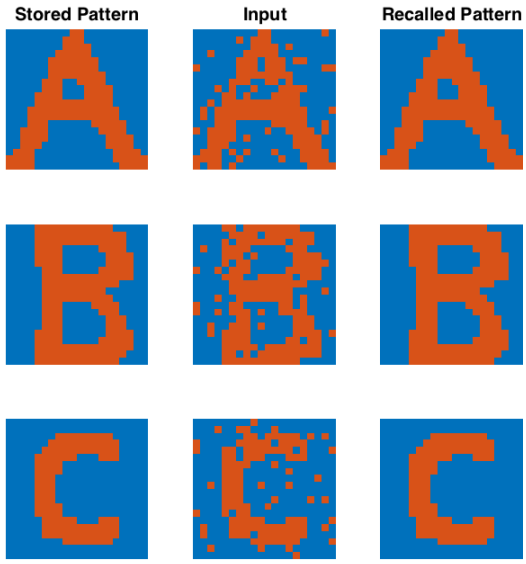

Figure 2: Example recall with SAM. After three patterns were stored, the network could restore noisy versions of these patterns.

3 Results

This section describes the theoretical computation of SAM's retrieval performance, memory efficiency, and simulation results.

3.1 Theoretical Performance

In the theory section, we focus on the retrieval performance of already stored patterns. For correct retrieval, we expect a pattern x to be recalled after a single iteration pass through the network, i.e., $f(x) = x$, because only then the pattern is stable. Other possible outcomes of $f(x)$ are zero activation, activating the wrong pattern or an intersection (for inhibition) or union (without inhibition) of patterns. When activating an intersection, it is still possible that a second iteration step will yield the correct pattern, but then the third iteration will revert back to the intersection because $f(x)$ *was* the intersection (which is rare but can happen). Therefore, we are interested in the probability of $f(x) = x$ for stored patterns.

Here, we assume that the training patterns are probabilistically independent. After storing a training pattern that activates m neurons, the probability that the same pattern activates the correct hidden neuron is

$$p_c(m) = 1 - \sum_{k=0}^{\theta-1} \binom{m}{k} p_S^k (1 - p_S)^{m-k} \quad . \tag{3}$$

The same pattern, however, may also activate a wrong hidden neuron of another stored pattern. To compute this probability, consider, first, the number b of neurons that overlap between our pattern and another pattern (Figure 3). The probability of having b overlapping neurons is computed as

$$p_o(b) = \frac{\binom{m}{b}\binom{n-m}{m-b}}{\binom{n}{m}} \quad , \tag{4}$$

where $\binom{m}{b}$ is the number of possibilities that b neurons fall inside the wrong pattern and $\binom{n-m}{m-b}$ the number of possibilities that $m - b$ neurons fall outside the wrong pattern. For $p_o(b)$, we can theoretically derive an upper bound that more easily shows the

dependence on n and m. Writing the factorial terms in product form, we observe

$$p_o(b) < \frac{1}{b!}\left(\frac{m^2}{n-2m+b}\right)^b \quad , \tag{5}$$

assuming $n > m$. That is, with increasing n, the probability of an overlap between two patterns decreases and gets arbitrarily close to zero.

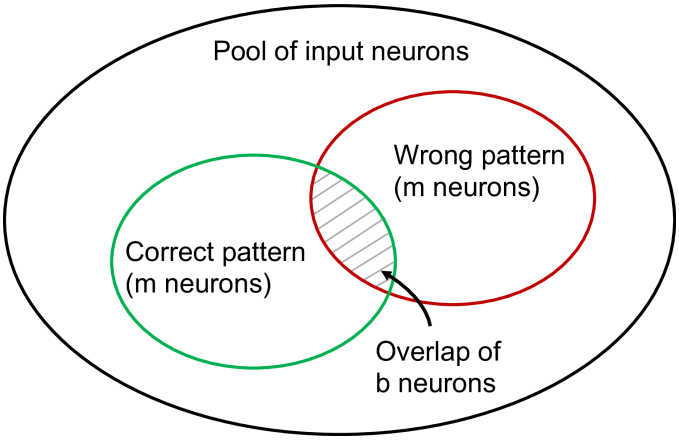

Figure 3: Neurons between stored patterns may overlap.

Each activated overlapping neuron is connected with probability p_S to the hidden neuron of the wrong pattern. So, the probability p_w to activate a hidden neuron of a specific wrong pattern is

$$p_w(m) = \sum_{b=\theta}^{m} p_o(b)p_c(b) \quad . \tag{6}$$

For the following computation of correct-retrieval probabilities, we consider the with and without inhibition cases separately. When using inhibition, we get a correct pattern retrieval if more hidden neurons of that pattern are activated than all other hidden neurons combined because here, the inhibition suppresses all the positives activations of other patterns.

Assume we have h hidden neurons per pattern, then the probability, p_+, that a given number j of hidden neurons of the correct pattern is active is

$$p_+(j) = \binom{h}{j} p_c(m)^j (1 - p_c(m))^{h-j} \quad . \tag{7}$$

On the other hand, the probability of activating less than j undesired hidden neurons is

$$p_-(j) = \sum_{l=0}^{j-1} \binom{(N-1)h}{l} p_w(m)^l \left(1 - p_w(m)\right)^{(N-1)h-l} \quad . \tag{8}$$

We obtain correct retrieval if the majority of activated hidden neurons belong to the correct pattern, resulting in the final probability of correct retrieval,

$$p_{\text{Inhib}} = \sum_{j=1}^{h} p_+(j) p_-(j) \quad . \tag{9}$$

This theoretical probability for correct retrieval is in good agreement with experiments (Figure 4). In these experiments, we stored into SAM various numbers of random patterns and tested the retrieval performance for the same patterns. We repeated each experiment 10 times.

Without inhibition, we get a correct retrieval if at least one correct hidden neuron is active, but no other hidden neuron gets activated. As a result, we obtain

$$p_{\text{NoInhib}} = \left(1 - (1 - p_c(m))^h\right) \left(1 - p_w(m)\right)^{(N-1)h} \quad . \tag{10}$$

Given these theoretical results on the retrieval performance, we can evaluate its dependence on the parameters, particularly, the number of training patterns, number of input and hidden neurons, number of active neurons in a pattern (pattern size m), probability of synaptic connection, p_S, and activation threshold, θ. In the following, we compare SAM with and without inhibition.

The retrieval performance degrades when increasing the number of stored patterns (Fig. 4c,d), and a larger number of input neurons enables a larger memory capacity. For example, for $n = 2000$, 1000 patterns can be reliably recalled, $p_{\text{Inhib}} \approx 99.97\%$. Inhibition improves the storage capacity.

Increasing the number of hidden neurons, h, improves the retrieval performance if n is sufficiently large (Fig. 5a,b). With inhibition, when increasing n, the transition from $p_{\text{Inhib}} = 0$ to $p_{\text{Inhib}} = 1$ approaches a step function in the limit $h \to \infty$.

The optimal firing threshold, θ, depends on the size, m, of a stored pattern and the probability of synaptic connections, p_S. For a given threshold, if m is too small, then a hidden neuron cannot be activated and retrieval fails. If, on the other hand, m is too large then the chance of activating a wrong pattern is large and retrieval can fail again. The parameters m, p_S, and θ are closely linked. For best retrieval, there is an optimal ratio of θ and the average number of connections per pattern, which is $m * p_S$ (Fig. 5c,d). With inhibition, this optimal ratio is about 0.6. We used this ratio to compute θ for all other tests and experiments.

3.2 Memory Efficiency

In this section, we investigate the memory efficiency. The efficiency, η, of a network is the total number of bits of stored patterns divided by the actual number of bits required for storage. Here, each pattern consists of n bits. In SAM, each pattern activates a subset of m input neurons; so, we need mhp_S forward and nh backward connections (when including the inhibitory connections). To encode each forward connection, we need $\log_2 n + \log_2 h$ bits to identify the connecting neurons. For the backward connections,

we need a $n \times h$ binary matrix with entries +1 or -1. As a result, our efficiency is

$$\eta = \frac{n}{mhp_S(\log_2 n + \log_2 h) + nh} \quad . \tag{11}$$

For example, for $n = 2000$, $h = 2$, $m = 200$, and $p_S = 0.1$, we obtain $\eta \approx 0.447$.

In contrast, for a symmetric Hopfield network, we require $n(n-1)/2$ connection weights, and for each connection, we need $\log_2(n/(2\log n)+1)$ bits (Gripon & Berrou, 2011), where $\log$ is the natural logarithm. Here, all patterns are stored in the same weight matrix. To compute the efficiency, we need to consider the storage capacity, which has an upper limit of $n/(2\log n)$ (McEliece et al., 1987). With this number, we store $n^2/(2\log n)$ bits, and the efficiency becomes

$$\eta_H = \frac{n}{(n-1)\log n \log_2(n/(2\log n)+1)} \quad , \tag{12}$$

e.g., for $n = 2000$, $\eta_H \approx 0.0187$ — SAM had a $24\times$ larger value. If the size of a pattern, m, is bounded, the efficiency of SAM scales favorable with the network size, $\eta = 1/h$ in the limit $n \to \infty$. In contrast, the efficiency of the Hopfield network decreases to zero; $\eta_H = 0$ in the limit $n \to \infty$ because $n_H < 1/\log n$ for sufficiently large n. This behavior even holds with improvements like Storkey's rule (Storkey, 1997), where $n_H < 1/\sqrt{\log n}$ for sufficiently large n.

3.3 Simulation Results

This section shows the robustness of SAM with respect to recalling perturbed patterns, the effect of the recurrent dynamics on spurious memories, and comparisons against the Hopfield network. In all experiments, we used the following settings for SAM: $h = 2$

hidden neurons, probability of connection, $p_S = 0.1$, and activation threshold for the hidden neurons, $\theta = 0.6mp_S$.

To test for robustness, we stored 1000 random patterns with $m = 200$ in a SAM with $n = 2000$ input neurons. Then, we perturbed each of the training patterns by randomly selecting a given number of neurons (size of perturbation). For each selected neuron, were altered its state, i.e., making it active if inactive and inactive if active. Figure 6b shows an example of a perturbed pattern inverting 100 neurons (half the number of active neurons in a pattern). To calculate the probability of correct retrieval, we tested each perturbed pattern, iterated SAM for five steps (one step includes one forward and backward pass), and repeated the entire experiment 10 times.

With inhibition, SAM converged after only two iterations. The change in probability for correct retrieval was small between the first and second iteration (< 0.1 percentage points for perturbations ≤ 100 with a maximum of about 0.6 points at 300). For perturbations of up to about 100 neurons, the probability for correct retrieval remained high (99.4%) when using inhibition (Fig. 6a). Without inhibition, convergence occurred after one iteration, but the retrieval performance was lower.

We tested the impact of the recurrent dynamics on the probability of spurious memories. In each experimental run, we stored 1000 random training patterns, and then tested the network on 1000 new random patterns. For both sets of patterns, the pattern size was $m = 100$. For pattern recall, we iterated SAM for ten steps. We averaged the results from 100 runs for each setting. A spurious memory was a pattern that differed from all stored patterns and consisted of at least one active neuron.

As a result, when using inhibition, the probability of spurious activations dropped

with each iteration step until convergence, which happened after seven steps (Fig. 7a). Without inhibition, however, the spurious activity got worse after the first iteration step and converged after the second (Fig. 7b).

For the comparison with the Hopfield network, we stored again random patterns with $m = 100$. We varied the number of training patterns and number of input neurons, n. For the Hopfield network, we used n fully-connected neurons. Because of the hidden neurons, our network used more neurons in total than the Hopfield network, but still required less information per stored pattern for the same number n (see Section 3.2). We tested the networks, first, on perturbed training patterns; here, the perturbation size was 5. Second, we tested the networks on 1000 newly generated random patterns to probe for spurious memories. For SAM, in the first test, we used only one iteration step for recall because one step was sufficient for a small perturbation (see above). In the second test, we used five iteration steps at which point the network reached almost the final state at convergence (Fig. 7a).

To train and test the Hopfield network, we used the MATLAB$^{\text{TM}}$ Neural Networks Toolbox (Version 8.4 from the Release 2015b), particularly, the `newhop` function for training and the `net` function with five relaxation steps for recall. `newhop` is based on Li's model (Li et al., 1989) and creates a fully-connected symmetric Hopfield network. We repeated all experiments comparing SAM and Hopfield networks ten times (including training-set generation) to obtain means and standard deviations.

On the perturbed training patterns, the performance of SAM degraded more gracefully compared to Hopfield when increasing the number of stored patterns (Fig. 8a,b). Moreover, when storing 1000 patterns, we could recover a near optimal performance

(99.4% correct retrieval) with SAM when increasing n to 2000, while the Hopfield network remained at 0% correct retrieval.

When presenting newly generated random patterns, the Hopfield network generated spurious memories in all cases (Fig. 8c,d). In contrast for SAM, the probability of spurious memories could be arbitrarily small with sufficiently large n relative to the number of stored pattern, e.g., $< 10^{-4}$ for $n = 1500$ and $N = 1000$ (Fig. 8c,d). Moreover, for large n, SAM converged in a single integration step (e.g., for $n = 2000$ and $N = 1000$). With smaller n compared to N, occasionally more iteration steps were required: e.g., for $n = 1000$ and $N = 1000$, convergence occurred after 1 step for 66% of patterns, after 2 steps for 92% of patterns, and after 5 steps for 99.9% of patterns (the network converged when $f(x) = x$). With inhibition, the probability of spurious activations was more than an order of magnitude smaller than without inhibition.

4 Discussion

We investigated theoretically and in simulations a new computational model of associative memory, which is based on sparsely subsampling neural patterns (Hawkins & Ahmad, 2016). This model, SAM, can restore perturbed patterns, while being memory efficient and having a low probability of spurious memories. We used simple neural dynamics, assuming only the most basic functionality, namely, a neuron integrates simultaneously arriving spikes and fires if the spikes add up to a value above threshold. Moreover, we assumed Hebbian learning for storing memories.

SAM is a suitable model of associative memory for brain-sized networks. Particu-

larly, for large n, the retrieval performance was best; memory efficiency was best; spurious memories had the lowest probability, and the computational speed is independent of n. For a single iteration step, the computational complexity is $O(mhN)$. In contrast, a fully-connected network, like the Hopfield network, is unrealistic for brain-sized networks because it would require too many connections, $O(n^2)$. For the fully-connected Hopfield network, the computational complexity is $O(n^2)$ per iteration step for recall because the neural activity is multiplied by the $n \times n$ weight matrix.

Here, for comparison, we considered only fully-connected Hopfield networks, which are the most common, but researchers studied also variants on sparsely-connected networks (Kim et al., 2017; Folli et al., 2018). These works show evidence that the sparsity improves memory capacity but also come with caveats: Kim et al. (2017) report an increase in retrieval errors, and Folli et al. (2018) count attractors instead of the actual stored memories and thereby also include spurious activations in their capacity count. Suppressing spurious activations is the main advantage of SAM over Hopfield networks.

In SAM, the sparsity of connections helps improve memory efficiency and allows low activation thresholds. We found empirically that the optimal firing threshold, θ, given the probability of sparse connections, p_S, and pattern size, m, is $\theta \approx 0.6 p_S m$. If p_S would be 1, i.e., not sparse, then the activation threshold would be too high to be realistic. Our backward connections were not sparse, but the activation threshold was only 1 for the input neurons; so, we avoided the problem of unrealistically high activation thresholds.

As an alternative model, we could have used also sparse backward connections, which would have to be combined with more hidden neurons, such that their combined

activity activates an entire stored pattern. Here, the activation threshold for the input neurons would need to be higher and scale with the number h of hidden neurons. We leave the investigation of this alternative model for future work.

We found that inhibition helps improving storage capacity and robustness against perturbations and lowers the probability of spurious memories. This result is consistent with previous findings showing the benefit of inhibition in the brain (Caspary et al., 2008; Müllner et al., 2015; Barron et al., 2017). In biological networks, inhibition is more localized than in SAM (Müllner et al., 2015; Barron et al., 2017). For simplicity, we formed inhibitory connections between hidden neurons and *all* input neurons that were not part of a pattern. Alternatively, biologically more realistic is having only selected/localized inhibitory connections. This setting would result in a retrieval performance that lies between our two results for inhibition and without it, but with the benefit of having a better memory efficiency: $\eta \propto n/\log n$ for large n.

The main benefit of the recurrent dynamics of SAM with inhibition was to reduce spurious memories. Here, spurious memories happen to be intersections of stored patterns. Applying the network dynamics, $f(x)$, on a random input pattern can result in an intersection of two or more stored patterns. Applying $f(x)$ again may result in the same intersection, further shrink it (i.e., an intersection of fewer patterns), or reduce it to zero. In this way, the recurrent dynamics reduces the probability of spurious memories. Without inhibition, however, the applying $f(x)$ twice increased the probability of spurious memories. Interestingly, after the first iteration step, the probability of spurious activation was about the same for inhibition and without it. The reason is that the probability for obtaining an intersection of patterns or a union (as in the case without

inhibition) is about the same after one step because two independent random patterns of 100 active out of 1000 neurons overlap in at least one neuron with probability close to 1 ($p = 0.99997$). Apart from spurious memories, the recurrent dynamics had only a minor impact on the probability of correct retrieval. When using perturbed patterns, SAM converged after two iterations, and the result after the second iteration was only slightly better than after the first.

Gibson & Robinson (1992) presented a sparse associative memory model that is related to our model. They use a recurrent neural network with sparse excitatory connections and an inhibitory neuron connected to all n input neurons. As in our work, a pattern consists of a set of active neurons from the pool of input neurons. Different from our work, however, there are no sparse connections to hidden neurons. This difference may result in a lower storage capacity: for example, for $n = 3000$ and pattern size $m = 300$, Gibson & Robinson (1992) report that only *50 patterns* can be stored simultaneously when using 2.25 million excitatory connections. In contrast, SAM can store at least *3000 patterns* with 100% correct retrieval using 1.98 million excitatory connections ($h = 2$).

Our network architecture, the separation between input and hidden layers, is similar to R-nets (Vogel, 1998), which use also two layers of neurons and sparse connections between them. There, however, the author uses symmetric connections, while our connections are asymmetric. In addition, there are differences in the dynamics: R-nets require more recall iterations (about 15 when storing around 60 patterns) and spurious neurons limit storage capacity (Vogel, 1998).

In summary, sparsely connecting to neural patterns, as described by Hawkins &

Ahmad (2016), can give rise to a competitive associative memory model that solves the problem of spurious memories. This new memory model, SAM, may lack biological details, but it can be a key model element for neural architectures that mimic brain-like functions or biological computational principles without excessively reproducing the details that are still poorly understood. In future work, we plan to expand SAM to build a model that explains the Aha Moment (Kounios et al., 2008).

References

Barron, H. C., Vogels, T. P., Behrens, T. E., & Ramaswami, M. (2017). Inhibitory engrams in perception and memory. *Proceedings of the National Academy of Sciences*, *114*(26), 6666–6674.

Barth, A. L., & Poulet, J. F. (2012). Experimental evidence for sparse firing in the neocortex. *Trends in Neurosciences*, *35*(6), 345-355.

Bruck, J., & Roychowdhury, V. P. (1990). On the number of spurious memories in the Hopfield model. *IEEE Transactions on Information Theory*, *36*(2), 393-397.

Caspary, D. M., Ling, L., Turner, J. G., & Hughes, L. F. (2008). Inhibitory neurotransmission, plasticity and aging in the mammalian central auditory system. *Journal of Experimental Biology*, *211*(11), 1781–1791.

Folli, V., Gosti, G., Leonetti, M., & Ruocco, G. (2018). Effect of dilution in asymmetric recurrent neural networks. *Neural Networks*, *104*, 50-59.

Gibson, W. G., & Robinson, J. (1992). Statistical analysis of the dynamics of a sparse associative memory. *Neural Networks*, *5*, 645-661.

Gripon, V., & Berrou, C. (2011). Sparse neural networks with large learning diversity. *IEEE Transactions on Neural Networks*, *22*(7), 1087-1096.

Hassoun, M. H. (1993). *Associative neural memories: Theory and implementation*. Oxford University Press.

Hawkins, J., & Ahmad, S. (2016). Why neurons have thousands of synapses, a theory of sequence memory in neocortex. *Frontiers in Neural Circuits*, *10*, 23.

Hoffmann, H., Howard, M. D., & Daily, M. J. (2011). Fast pattern matching with time-delay neural networks. In *International joint conference on neural networks.* IEEE.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the USA*, *79*, 2554-2558.

Hopfield, J. J. (2008). Searching for memories, sudoku, implicit check bits, and the iterative use of not-always-correct rapid neural computation. *Neural Computation*, *20*, 1119-1164.

Kim, D.-H., Park, J., & Kahng, B. (2017). Enhanced storage capacity with errors in scale-free hopfield neural networks: An analytical study. *PLOS One*, *12*(10), e0184683.

Kounios, J., Fleck, J. I., Green, D. L., Payne, L., Stevenson, J. L., Bowden, E. M., & Jung-Beeman, M. (2008). The origins of insight in resting-state brain activity. *Neuropsychologia*, *46*(1), 281-291.

Krotov, D., & Hopfield, J. J. (2016). Dense associative memory for pattern recognition. In *Conference on Neural Information Processing Systems.*

Lansner, A. (2009). Associative memory models: from the cell-assembly theory to biophysically detailed cortex simulations. *Trends in Neurosciences*, *32*(3), 178 - 186.

Li, J., Michel, A. N., & Porod, W. (1989). Analysis and synthesis of a class of neural networks: linear systems operating on a closed hypercube. *IEEE Transactions on Circuits and Systems*, *36*(11), 1405-1422.

Löwel, S., & Singer, W. (1992). Selection of intrinsic horizontal connections in the visual cortex by correlated neuronal activity. *Science*, *255*(5041), 209-212.

McEliece, R. J., Posner, E. C., Rodemich, E. R., & Venkatesh, S. S. (1987). The capacity of the Hopfield associative memory. *IEEE Transactions on Information Theory*, *33*(4), 461-482.

Minai, A. A., & Levy, W. B. (1993, Dec 01). The dynamics of sparse random networks. *Biological Cybernetics*, *70*(2), 177–187.

Mofrad, A. A., & Parker, M. G. (2017). Nested-clique network model of neural associativememory. *Neural Computation*, *29*, 1681-1695.

Müllner, F. E., Wierenga, C. J., & Bonhoeffer, T. (2015). Precision of inhibition: Dendritic inhibition by individual gabaergic synapses on hippocampal pyramidal cells is confined in space and time. *Neuron*, *87*(3), 576 - 589.

Storkey, A. (1997). Increasing the capacity of a hopfield network without sacrificing functionality. In W. Gerstner, A. Germond, M. Hasler, & J.-D. Nicoud (Eds.), *Artificial neural networks — ICANN'97* (pp. 451–456). Springer Berlin Heidelberg.

Vogel, D. D. (1998). Auto-associative memory produced by disinhibition in a sparsely connected network. *Neural Networks*, *11*, 987-908.

Wills, T. J., Lever, C., Cacucci, F., Burgess, N., & O'Keefe, J. (2005). Attractor dynamics in the hippocampal representation of the local environment. *Science*, *308*(5723), 873–876.

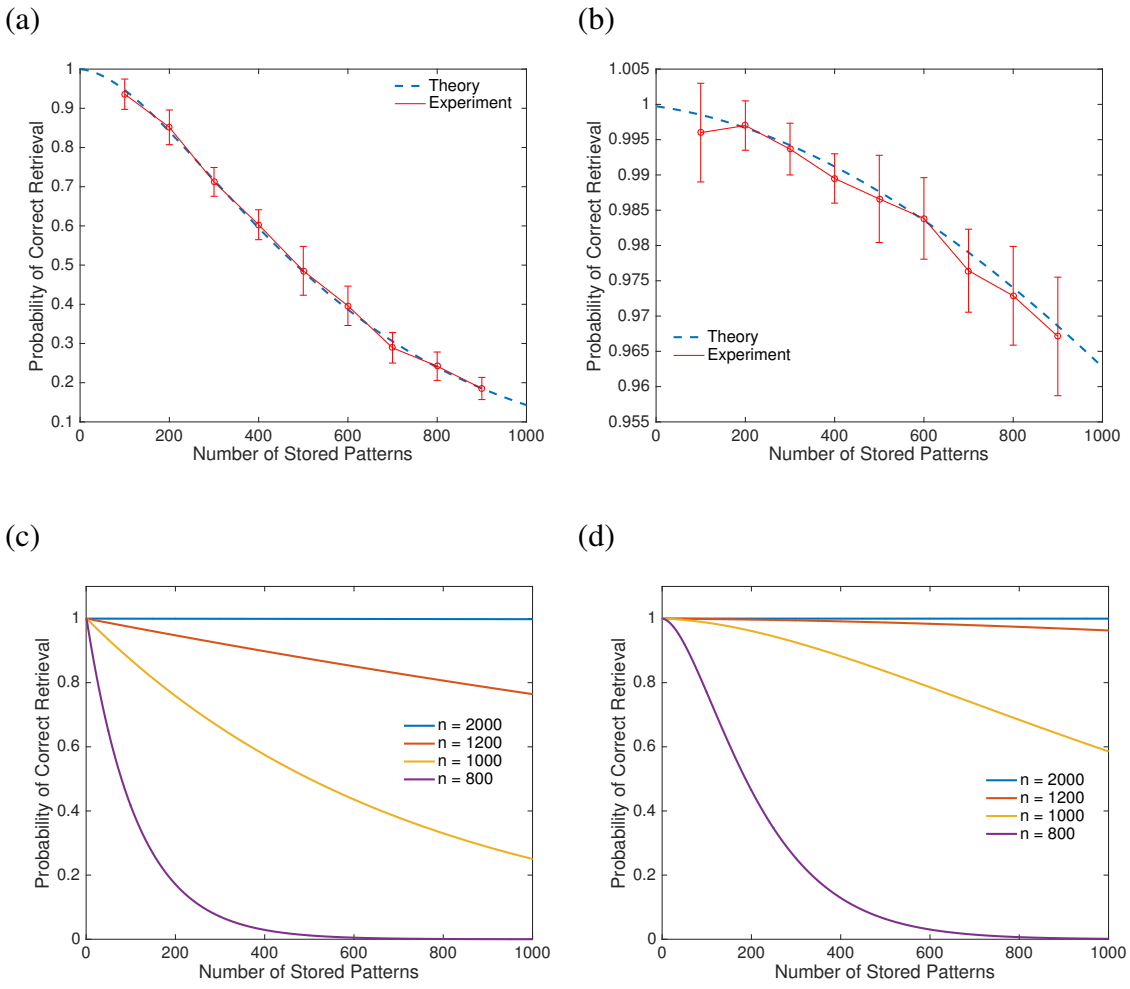

Figure 4: Probability of correct pattern recall depending on the number of stored patterns. Theoretical predictions matched experimental values for $n = 900$ (a) and $n = 1200$ (b). Experimental data show mean $\pm$ SD. The retrieval performance depends on the number of input neurons, shown for SAM without (c) and with inhibition (d). For all results in this figure, $h = 2$, $m = 200$, $p_S = 0.1$, and $\theta = 12$.

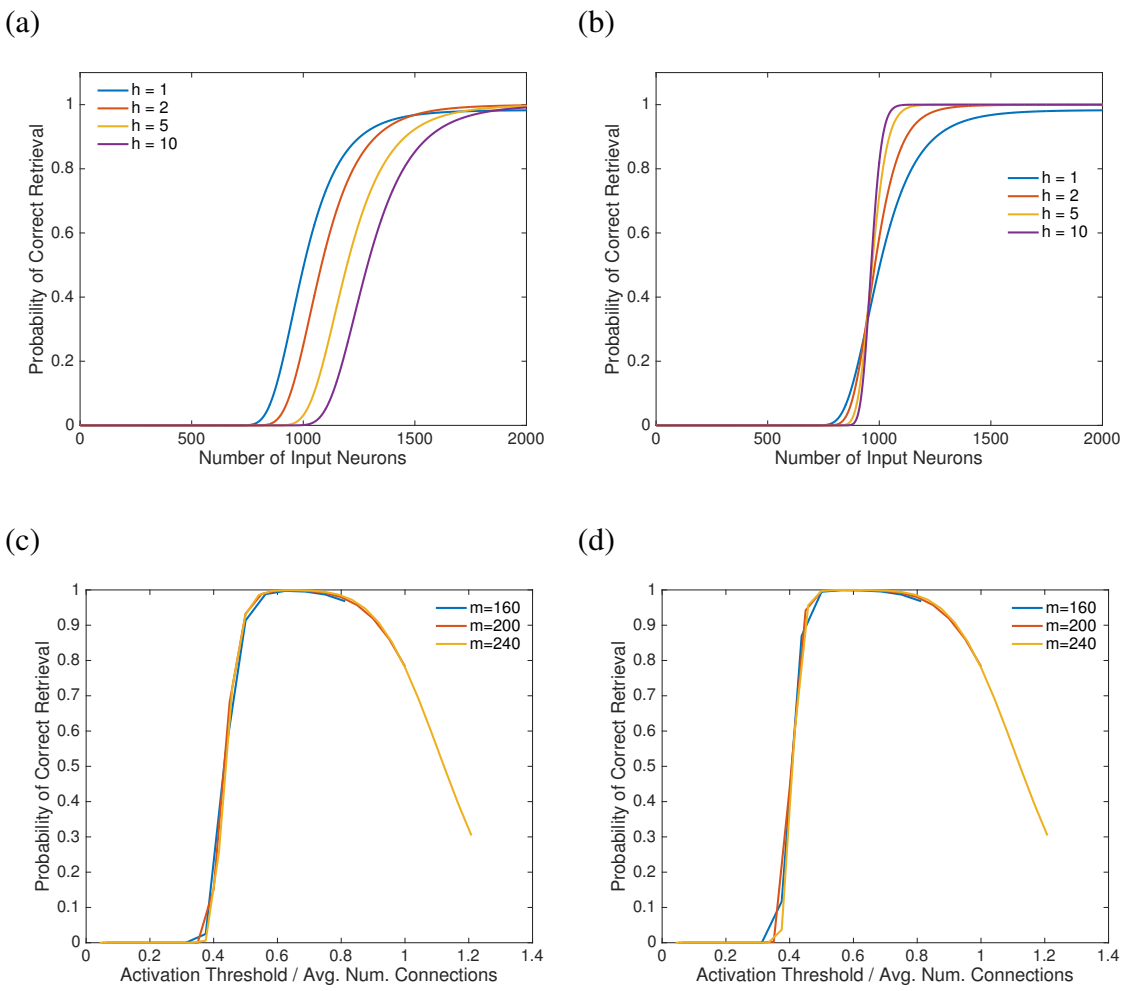

Figure 5: Probability of correct pattern recall depending on network parameters for storing 1000 patterns, showing dependence on the number of input, n, and hidden neurons, h, for SAM without (a) and with inhibition (b). Here, $m = 200$, $p_S = 0.1$, and $\theta = 12$. In addition, recall depends on the normalized firing threshold, $\theta/(mp_S)$, showing results for SAM without (c) and with inhibition (d). Here, using $n = 2000$, $h = 2$, and $p_S = 0.1$.

(a) (b)

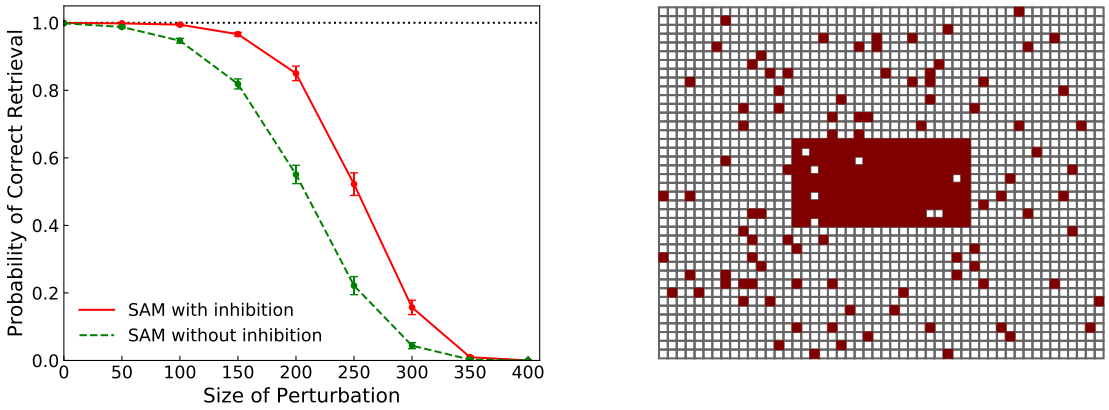

Figure 6: Robustness to perturbation, showing probability of correct pattern recall with inhibition and without inhibition depending on size of perturbation (a) and example perturbation of size 100 of a rectangular neural activation pattern (b). Here, the experiments used 1000 **stored patterns,** $n = 2000$**,** $h = 2$**,** $m = 200$**,** $p_S = 0.1$**, and** $\theta = 12$**. Data show mean** $\pm$ **SD.**

(a) (b)

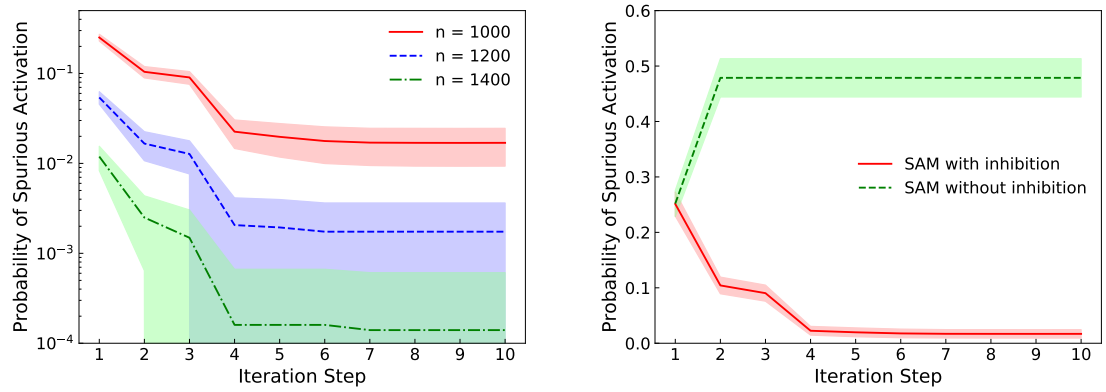

Figure 7: Recurrent dynamics in SAM reduce spurious activations when using inhibition (a) but increase them without inhibition (b). Here, the experiments used 1000 **stored patterns,** $n = 1000$ **(unless specified otherwise),** $h = 2$**,** $m = 100$**,** $p_S = 0.1$**, and** $\theta = 6$**, and data show mean** $\pm$ **SD.**

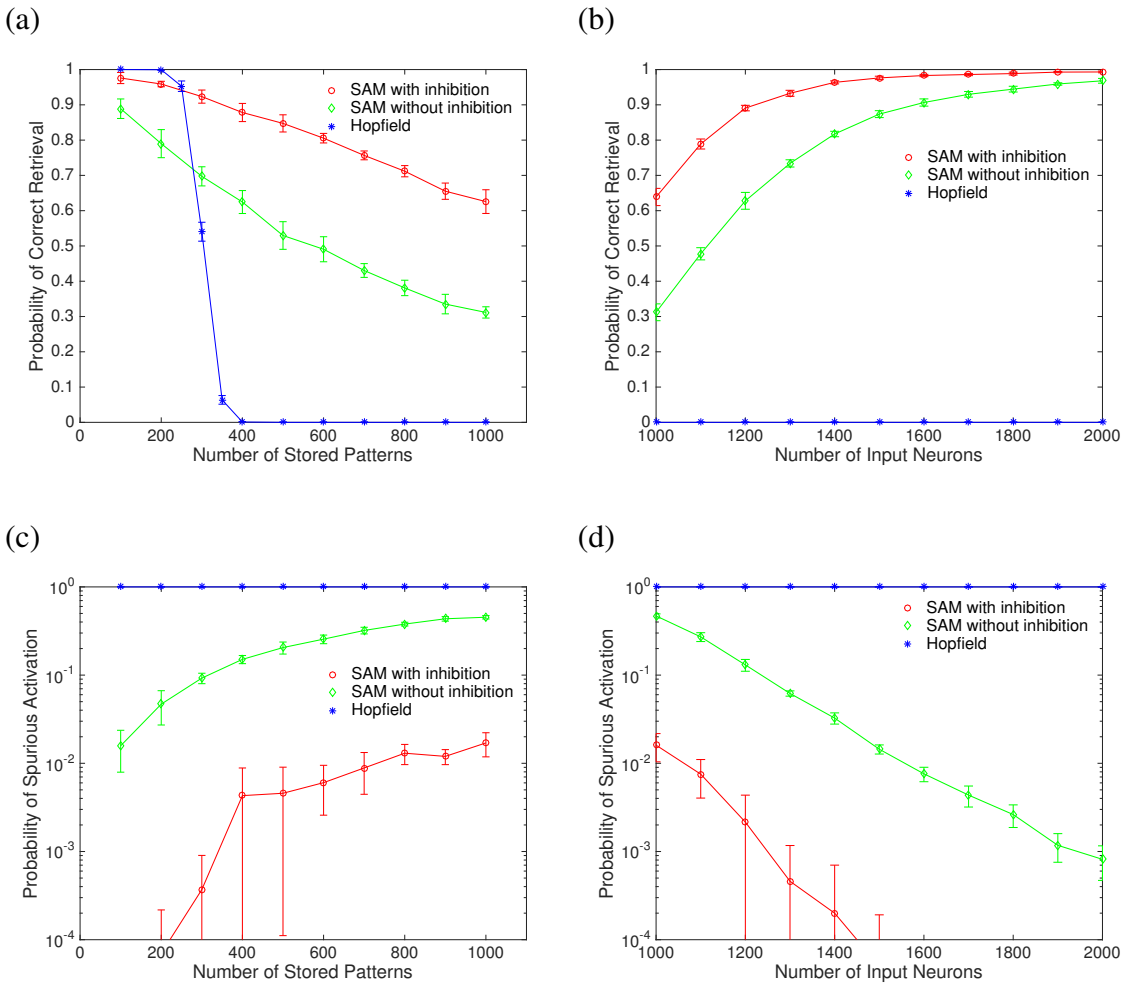

Figure 8: Comparison between SAM and the Hopfield network. First, comparing probability of correct pattern retrieval as a function of the number of stored pattern when $n = 1000$ (a) and as a function of the number of input neurons, n, when storing 1000 patterns (b). Here, the perturbation size was 5. Second, comparing the probability of spurious memories as a function of the number of stored pattern when $n = 1000$ (c) and as a function of the number of input neurons, n, when storing 1000 patterns (d). Parameters were $h = 2$, $m = 100$, $p_S = 0.1$, and $\theta = 6$, and data show mean $\pm$ SD.