Kernel Memory Networks: A Unifying Framework for Memory Modeling

Georgios Iatropoulos[1,2] **Johanni Brea**[1,*] **Wulfram Gerstner**[1,*]

{firstname.lastname}@epfl.ch
[1]Laboratory of Computational Neuroscience, [2]Blue Brain Project
École Polytechnique Fédérale de Lausanne
Switzerland

Abstract

We consider the problem of training a neural network to store a set of patterns with maximal noise robustness. A solution, in terms of optimal weights and state update rules, is derived by training each individual neuron to perform either kernel classification or interpolation with a minimum weight norm. By applying this method to feed-forward and recurrent networks, we derive optimal models, termed kernel memory networks, that include, as special cases, many of the hetero- and auto-associative memory models that have been proposed over the past years, such as modern Hopfield networks and Kanerva's sparse distributed memory. We modify Kanerva's model and demonstrate a simple way to design a kernel memory network that can store an exponential number of continuous-valued patterns with a finite basin of attraction. The framework of kernel memory networks offers a simple and intuitive way to understand the storage capacity of previous memory models, and allows for new biological interpretations in terms of dendritic non-linearities and synaptic cross-talk.

1 Introduction

Although the classical work on attractor neural networks reached its peak in the late 1980's, with the publication of a number of seminal works [e.g., 2, 19, 21, 25], recent years have seen a renewed interest in the topic, motivated by the popularity of the attention mechanism [63], external memory-augmented neural networks [23, 64], as well as a new generation of energy-based attractor networks models, termed modern Hopfield networks (MHNs), capable of vastly increased memory storage [16, 34]. Recent efforts to understand the theoretical foundation of the attention mechanism have, in fact, shown that it can be linked to Hopfield networks [35, 55], but also to Kanerva's sparse distributed memory (SDM) [8, 29], and to the field of kernel machines [61, 66]. The last connection is particularly intriguing, in light of the many theoretical commonalities between neural networks and kernel methods [9, 10, 27, 46, 65]. Overall, these results suggest that a unified view can offer new insights into memory modeling and new tools for leveraging memory in machine learning.

In this work, we aim to clarify some of the overlap between the fields of memory modeling and statistical learning, by integrating and formalizing a set of theoretical connections between Hopfield networks, the SDM, kernel machines, and neuron models with non-linear dendritic processing.

[*]Joint senior authors.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

1.1 Our contribution

- We derive a set of normative kernel-based models that describe the general mathematical structure of feed-forward (i.e., hetero-associative) and recurrent (i.e., auto-associative) memory networks that can perform error-free recall of a given set of patterns with maximal robustness to noise.
- We show that the normative models include, as special cases, the classical and modern Hopfield network, as well as the SDM.
- We derive a simple attractor network model for storing an exponential number of continuous-valued patterns with a finite basin of attraction. We discuss its similarity to attention.

Furthermore, we explain how classifiers with non-linear kernels can be interpreted as general forms of neuron models with non-linear dendritic activation functions and synaptic cross-talk.

1.2 Related work

Our work is primarily related to [8, 34, 35, 44, 55]. While MHNs are extensively analyzed in [34, 35, 44, 55], the approach is energy-based and makes no statements about the relation between MHNs and kernel methods; a brief comment in [55] mentions some similarity to SVMs, but this is not further explained. The work by [8] focuses on the SDM and its connection to attention. It observes that the classical Hopfield network is a special case of the SDM, but no further generalization is made, and kernel methods are not mentioned. In our work, we place MHNs and the SDM in a broader theoretical context by showing that *both* models are special suboptimal cases of a family of memory networks that can be derived with a normative kernel-based approach.

2 Background

Consider the following simple model of hetero-associative memory: a single-layer feed-forward network consisting of a single output neuron connected to N_{in} inputs with the weights $\mathbf{w} \in \mathbb{R}^{N_\phi}$. The output $s_{\text{out}} \in \{\pm 1\}$ is given by

$$s_{\text{out}} = \text{sgn}\Big[\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{s}_{\text{in}}) - \theta\Big] \tag{1}$$

where $\mathbf{s}_{\text{in}}$ is the input vector (also called query), θ the threshold, and $\boldsymbol{\phi}$ a function that maps the "raw" input to a N_ϕ-dimensional feature space, where typically $N_\phi \gg N_{\text{in}}$. Suppose that we are given a set of M input-output patterns $\{\boldsymbol{\xi}^\mu_{\text{in}}, \xi^\mu_{\text{out}}\}_{\mu=1}^M$, in which every entry ξ is randomly drawn from $\{\pm 1\}$ with sparseness $f := \mathbb{P}(\xi = 1)$. In order for the neuron to store the patterns in a way that maximizes the amount of noise it can tolerate while still being able to recall all patterns without errors, one needs to find the weights that produce the output ξ^μ_{out} in response to the input $\boldsymbol{\xi}^\mu_{\text{in}}$, $\forall \mu$, and that maximize the smallest Euclidean distance between the inputs and the neuron's decision boundary. Using Gardner's formalism [19, 21], this problem can be expressed as

$$\begin{aligned} \arg\max_{\mathbf{w}} \ \kappa \quad \text{s.t.} \quad & \xi^\mu_{\text{out}}\Big(\mathbf{w}^\top \boldsymbol{\phi}(\boldsymbol{\xi}^\mu_{\text{in}}) - \theta\Big) \geq \kappa, \ \forall \mu \\ & \|\mathbf{w}\|_2 = \bar{w} \end{aligned} \tag{2}$$

where $\bar{w} > 0$ is a constant. This is equivalent to solving

$$\min_{\mathbf{w}} \|\mathbf{w}\|_2 \quad \text{s.t.} \quad \xi^\mu_{\text{out}}\Big(\mathbf{w}^\top \boldsymbol{\phi}(\boldsymbol{\xi}^\mu_{\text{in}}) - \theta\Big) \geq 1, \ \forall \mu \tag{3}$$

which can be directly identified as the support vector machine (SVM) problem for separable data [12]. The solution to Eq. 3 can today be found in any textbook on basic machine learning methods, and yields an optimal output rule that can be written in a *feature* and *kernel* form

$$s_{\text{out}} = \text{sgn}\left[\sum_\mu^M \alpha^\mu \xi^\mu_{\text{out}} \boldsymbol{\phi}(\boldsymbol{\xi}^\mu_{\text{in}})^\top \boldsymbol{\phi}(\mathbf{s}_{\text{in}}) - \theta\right] = \text{sgn}\left[\sum_\mu^M \alpha^\mu \xi^\mu_{\text{out}} K(\boldsymbol{\xi}^\mu_{\text{in}}, \mathbf{s}_{\text{in}}) - \theta\right] \tag{4}$$

where we, in the latter expression, have used the "kernel-trick" $K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\phi}(\mathbf{x}_i)^\top \boldsymbol{\phi}(\mathbf{x}_j)$. The solution depends on the Lagrange coefficients $\alpha^\mu \geq 0$, many of which are typically zero. Patterns with $\alpha^\mu > 0$ are called *support vectors*.

3 Kernel memory networks for binary patterns

3.1 *Hetero*-associative memory as a feed-forward SVM network

We begin by considering a hetero-associative memory network with an arbitrary number N_{out} output neurons, whose combined state we denote $\mathbf{s}_{\text{out}}$. In order for the network as a whole to be able to tolerate a maximal level of noise and still successfully recall its stored memories, we solve Eq. 3 for each neuron independently. As each neuron can have a different classification boundary along with a different set of support vectors, its weights will, in general, be characterized by an independent set of M Lagrange coefficients. To simplify the notation, we represent these coefficients α_i^μ, across neurons i and patterns μ, as entries in the matrix $\mathbf{A}$, where $(\mathbf{A})_{i\mu} = \alpha_i^\mu$. We also combine all thresholds in the vector $\boldsymbol{\theta} = (\theta_1, \dots, \theta_{N_{\text{out}}})$, and all input and output patterns as columns in the matrices $\mathbf{X}_{\text{in}} = (\boldsymbol{\xi}_{\text{in}}^1, \dots, \boldsymbol{\xi}_{\text{in}}^M)$ and $\mathbf{X}_{\text{out}} = (\boldsymbol{\xi}_{\text{out}}^1, \dots, \boldsymbol{\xi}_{\text{out}}^M)$. Finally, we assume that all neurons have the same feature map, so that $\boldsymbol{\phi}_i = \boldsymbol{\phi}, \forall i$ (see Fig. 1). All functions are applied column-wise when the argument is a matrix, for example $\boldsymbol{\phi}(\mathbf{X}_{\text{in}}) = (\boldsymbol{\phi}(\boldsymbol{\xi}_{\text{in}}^1), \dots, \boldsymbol{\phi}(\boldsymbol{\xi}_{\text{in}}^M))$. The optimal response of the network can now be compactly summarized as follows.

Property 1 (Robust hetero-associative memory network)**.** *A single-layer hetero-associative memory network trained to recall the patterns $\mathbf{X}_{\text{out}}$ in response to the inputs $\mathbf{X}_{\text{in}}$ with maximal noise robustness, has an optimal output rule that can be written as*

$$\mathbf{s}_{\text{out}} = \operatorname{sgn}\Big[(\mathbf{A} \odot \mathbf{X}_{\text{out}})\boldsymbol{\phi}(\mathbf{X}_{\text{in}})^\top \boldsymbol{\phi}(\mathbf{s}_{\text{in}}) - \boldsymbol{\theta}\Big] \qquad \textit{(feature form)} \tag{5}$$

$$= \operatorname{sgn}\big[(\mathbf{A} \odot \mathbf{X}_{\text{out}})K(\mathbf{X}_{\text{in}}, \mathbf{s}_{\text{in}}) - \boldsymbol{\theta}\big] \qquad \textit{(kernel form)} \tag{6}$$

where $\odot$ denotes the Hadamard product.

3.2 *Auto*-associative memory as a recurrent SVM network

The hetero-associative network can be made auto-associative by setting $N_{\text{out}} = N_{\text{in}}$ and $\mathbf{X}_{\text{out}} = \mathbf{X}_{\text{in}}$. The network is now effectively recurrent, as each neuron can serve both as an input and output simultaneously (see Fig. 1). Consider a recurrent network with N neurons, whose state at time point t is denoted $\mathbf{s}^{(t)} \in \{\pm 1\}^N$, and whose dynamics evolve according to the update rule

$$s_i^{(t+1)} = \operatorname{sgn}\Big[\mathbf{w}_i^\top \boldsymbol{\phi}(\mathbf{s}^{(t)}) - \theta_i\Big] \tag{7}$$

where $\mathbf{w}_i \in \mathbb{R}^{N_\phi}$ is the weight vector to neuron $i = 1, \dots, N$. In order to make the patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^M$ fixed points of the network dynamics, we train each neuron i independently on every pattern μ to, again, produce the response ξ_i^μ when the rest of the network is initialized in $\boldsymbol{\xi}^\mu$. Moreover, we maximize the amount of noise that can be tolerated by the network while maintaining error-free recall by maximizing the smallest Euclidean distance between each neuron's decision boundary and its inputs. This maximizes the size of the attractor basins [17, 31]. The problem of training the entire network is, in this way, transformed into the problem of training N separate classifiers according to

$$\min_{\mathbf{w}_i} \|\mathbf{w}_i\|_2 \quad \text{s. t.} \quad \xi_i^\mu \Big(\mathbf{w}_i^\top \boldsymbol{\phi}(\boldsymbol{\xi}^\mu) - \theta_i\Big) \geq 1, \ \forall \mu, i \ . \tag{8}$$

The solution can be obtained by slightly modifying Property 1, and is stated below.

Property 2.1 (Robust auto-associative memory)**.** *A recurrent auto-associative memory network trained to recall the patterns $\mathbf{X}$ with maximal noise robustness has an optimal synchronous update rule that can be written as*

$$\mathbf{s}^{(t+1)} = \operatorname{sgn}\Big[(\mathbf{A} \odot \mathbf{X})\boldsymbol{\phi}(\mathbf{X})^\top \boldsymbol{\phi}(\mathbf{s}^{(t)}) - \boldsymbol{\theta}\Big] \qquad \textit{(feature form)} \tag{9}$$

$$= \operatorname{sgn}\Big[(\mathbf{A} \odot \mathbf{X})K(\mathbf{X}, \mathbf{s}^{(t)}) - \boldsymbol{\theta}\Big] \qquad \textit{(kernel form)} \tag{10}$$

Remark. With a linear feature map $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$, the optimal update is reduced to

$$\mathbf{s}^{(t+1)} = \operatorname{sgn}\Big[(\mathbf{A} \odot \mathbf{X})\mathbf{X}^\top \mathbf{s}^{(t)} - \boldsymbol{\theta}\Big] \tag{11}$$

where $(\mathbf{A} \odot \mathbf{X})\mathbf{X}^\top$ can be identified as the general form of the optimal weight matrix.

The solution described by Property 2.1 does not, in general, prohibit a neuron from having self-connections. Applying this constraint yields the following result.

Property 2.2 (Robust auto-associative memory without self-connections)**.** *A recurrent auto-associative memory network without self-connections, with the inner-product kernel $K(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i^\top \mathbf{x}_j)$, that has been trained to recall the patterns $\mathbf{X}$ with maximal noise robustness, has an optimal asynchronous update rule that can be written in the kernel form*

$$s_i^{(t+1)} = \operatorname{sgn}\left[\sum_{\mu}^{M} \alpha_i^\mu \xi_i^\mu k\left(\sum_{j\neq i}^{N} \xi_j^\mu s_j^{(t)}\right) - \theta_i\right] . \tag{12}$$

Storage capacity. An intuition for the storage capacity scaling of the hetero- and auto-associative memory networks can be gained by observing that the network as a whole will be able to successfully recall patterns as long as each neuron is able to correctly classify its inputs (or is very unlikely to produce an error). The capacity of the network can thereby be derived from the capacity of each individual neuron. It is well-known that a linear binary classifier can learn to correctly discriminate a maximum of $M_{\max} \approx 2D_{\mathrm{VC}}$ random patterns, where D_{VC} is the Vapnik-Chervonenkis dimension of the classifier [14, 19, 39, ch. 40]. For a neuron with N inputs and a linear feature map $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$, this results in $D_{\mathrm{VC}} = N$ and, thus, the capacity $M_{\max} \approx 2N$. Suppose, on the other hand, that the kernel is a homogeneous polynomial of degree p, so that $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^p$. In this case, $\boldsymbol{\phi}$ will contain all monomials of degree p composed of the entries in $\mathbf{x}$. As there are $\mathcal{O}(N^p)$ unique p-degree monomials (see Appendix A.1), the input dimensionality and $M_{\max}$ will be $\mathcal{O}(N^p)$. For the exponential kernel, which we can write as $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(\mathbf{x}_i^\top \mathbf{x}_j) = \sum_{p=0}^{\infty} (\mathbf{x}_i^\top \mathbf{x}_j)^p / p!$, the dimensionality of $\boldsymbol{\phi}$ will be $\sum_{p=0}^{N} \binom{N}{p} = 2^N$, which yields $M_{\max} \sim \mathcal{O}(e^N)$.

Special cases. In the following sections, we will show that many of the models of hetero- and auto-associative memory that have been proposed over the past years are special cases of the solutions in Properties 1, 2.1, and 2.2, characterized by specific choices of $\mathbf{A}$, $\boldsymbol{\phi}$, and K.

3.3 Kanerva's sparse distributed memory is a feed-forward SVM network

The sparse distributed memory (SDM), developed by Kanerva [29], is one of the most famous examples of a hetero-associative memory model. It has lately received much attention in the context of generative memory models [67] and attention layers in transformers [8].

The SDM consists of a register of N_ϕ memory slots, each associated with an address $\mathbf{z}_i \in \{\pm 1\}^{N_{\mathrm{in}}}$, $i = 1, \ldots, N_\phi$. All addresses are listed as rows in the matrix $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_{N_\phi})^\top$. The content of each slot is represented by an N_{out}-dimensional vector, initialized at zero. Suppose that we wish to store the M patterns $\mathbf{X}_{\mathrm{out}} = (\boldsymbol{\xi}_{\mathrm{out}}^1, \ldots, \boldsymbol{\xi}_{\mathrm{out}}^M)$ in the addresses $\mathbf{X}_{\mathrm{in}} = (\boldsymbol{\xi}_{\mathrm{in}}^1, \ldots, \boldsymbol{\xi}_{\mathrm{in}}^M)$, where all entries are random and bipolar. The basic idea of the SDM is to write the data to, and later read it from, multiple memory slots at once (hence the distributed storage); this ensures a degree of noise-robustness. In mathematical terms, the read-out of the SDM provided with a query $\mathbf{s}_{\mathrm{in}}$, is given by

$$\mathbf{s}_{\mathrm{out}} = \operatorname{sgn}\Big[\mathbf{X}_{\mathrm{out}}\, \Theta(\mathbf{Z}\mathbf{X}_{\mathrm{in}} - b)^\top\, \Theta(\mathbf{Z}\mathbf{s}_{\mathrm{in}} - b)\Big] \tag{13}$$

where Θ the Heaviside function with bias $b = N_{\mathrm{in}} - 2r$, and r is a parameter that determines the precision of the writing and reading process. Upon comparing Eqs. 13 and 5, the SDM can be directly identified as a special case of a suboptimal feed-forward SVM network in the feature form, with $\mathbf{A} = \mathbf{1}$, $\boldsymbol{\theta} = \mathbf{0}$, and the feature map $\boldsymbol{\phi}_{\mathrm{SDM}}(\mathbf{x}) = \Theta(\mathbf{Z}\mathbf{x} - b)$. When viewed as a kernel method, the function of the SDM is to store the dense addresses $\mathbf{X}_{\mathrm{in}}$ as sparse high-dimensional representations $\boldsymbol{\phi}_{\mathrm{SDM}}$, to make it easier to later determine the slots closest to a query $\mathbf{s}_{\mathrm{in}}$, and retrieve the relevant data.

Capacity. As the SDM is linear in $\boldsymbol{\phi}_{\mathrm{SDM}}$, with $D_{\mathrm{VC}} \approx N_\phi$, it follows from the analysis in Sec. 3.2 that one should expect the capacity to scale as $M_{\max} \sim \mathcal{O}(N_\phi)$. Moreover, one should expect a proportionality constant $\sim$0.1, since the SDM is suboptimal relative to the feed-forward SVM network, analogously to how the classical Hopfield network is suboptimal relative to the recurrent SVM network (see Sec. 3.4). This is consistent with earlier proofs [11, 30].

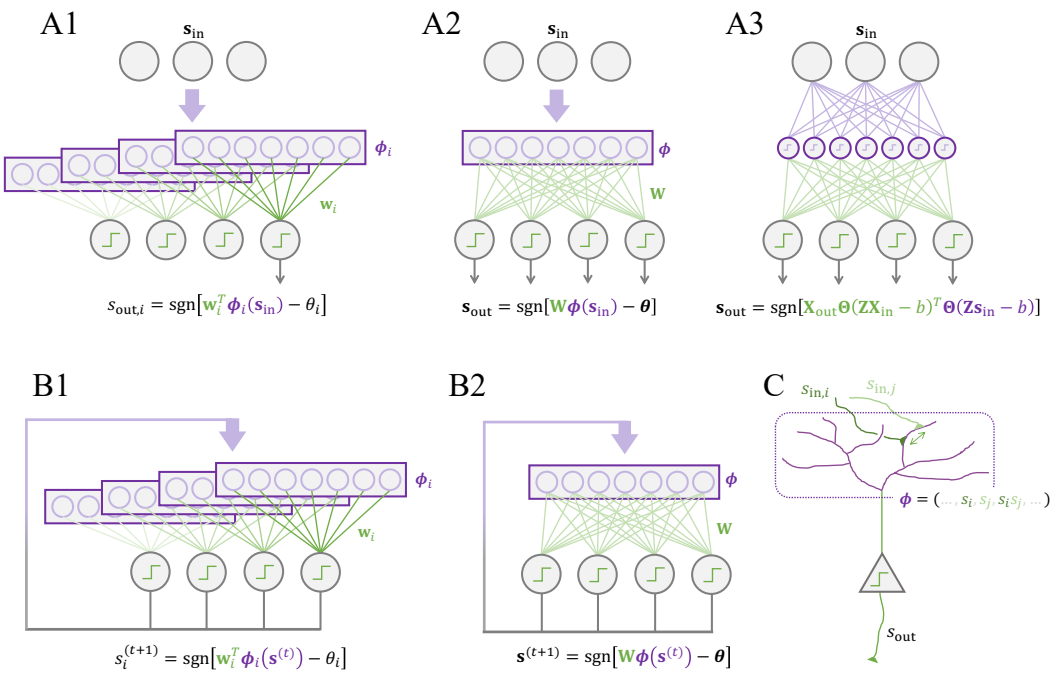

Figure 1: Graphical representation of (A1-A2) the feed-forward SVM network, (A3) the SDM, (B1-B2) the recurrent SVM network, and (C) an SVM mapped to the anatomy of a pyramidal cell (see Sec. 5). Circles represent neurons, while boxes represent the input transformation by the feature map $\boldsymbol{\phi}$, which can be dependent (A1, B1) or independent (A2, B2) of neuron index i.

Kernel of an infinite SDM. In practice, an SDM with a large number of memory slots N_ϕ requires calculations involving a large address matrix $\mathbf{Z}$. This can be avoided by applying the kernel-trick to Eq. 13 in the limit $N_\phi \to \infty$, which allows for the output to be computed with

$$\mathbf{s}_{\text{out}} = \text{sgn}\big[\mathbf{X}_{\text{out}} K_{\text{SDM}}(\mathbf{X}_{\text{in}}, \mathbf{s}_{\text{in}})\big] \tag{14}$$

where we have defined the kernel as

$$K_{\text{SDM}}(\mathbf{x}_i, \mathbf{x}_j) = \lim_{N_\phi \to \infty} \frac{\boldsymbol{\phi}_{\text{SDM}}(\mathbf{x}_i)^\top \boldsymbol{\phi}_{\text{SDM}}(\mathbf{x}_j)}{N_\phi} \tag{15}$$

in order to ensure convergence. In this section, we will derive this kernel for two different variants of the SDM and demonstrate that both are translation-invariant. It is interesting to note here that $\boldsymbol{\phi}_{\text{SDM}}$ is equivalent to a single-layer neural network with N_ϕ neurons, weights $\mathbf{Z}$, and bias b. This means that K_{SDM} is equivalent to the kernel of an infinitely wide neural network [10, 46, 65].

We begin by noticing that $\boldsymbol{\phi}_{\text{SDM}}(\mathbf{x})$ has a geometrical interpretation [8, 30]. It is a binary vector that indicates those memory addresses in $\mathbf{Z}$ that differ by at most r bits compared to $\mathbf{x}$. For any two bipolar vectors $\mathbf{z}$ and $\mathbf{x}$, the bit-wise difference can be computed as $\frac{1}{2}|\mathbf{z} - \mathbf{x}| = \frac{1}{4}\|\mathbf{z} - \mathbf{x}\|_2^2$. This means that $\boldsymbol{\phi}_{\text{SDM}}(\mathbf{x})$ indicates all addresses that lie within a sphere centered at $\mathbf{x}$ with radius $2\sqrt{r}$. Consequently, the inner product $\boldsymbol{\phi}_{\text{SDM}}(\mathbf{x}_i)^\top \boldsymbol{\phi}_{\text{SDM}}(\mathbf{x}_j)$ is the number of addresses located in the overlapping volume of two spheres centered at $\mathbf{x}_i$ and $\mathbf{x}_j$. Although an exact calculation of this quantity can be found in [8, 29], its connection to the SDM kernel has, to the best of our knowledge, not previously been made. We therefore modify the previously published expression with a normalization factor $1/2^{N_{\text{in}}}$ and state the following property.

Property 3.1 (Kernel of an infinite SDM on the hypercube)**.** *In the limit $N_\phi \to \infty$, the kernel of an SDM with N_ϕ memory slots, whose addresses are randomly drawn from $\{\pm 1\}^{N_{\text{in}}}$, is given by*

$$K_{\text{SDM}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2^{N_{\text{in}}}} \sum_{i=N_{\text{in}}-r-\lfloor\frac{\Delta}{2}\rfloor}^{N_{\text{in}}-\Delta} \sum_{j=[N_{\text{in}}-r-i]_+}^{\Delta-(N_{\text{in}}-r-i)} \binom{N_{\text{in}}-\Delta}{i} \cdot \binom{\Delta}{j} \tag{16}$$

where r is the bit-wise error threshold and Δ is the bit-wise difference between $\mathbf{x}_i$ and $\mathbf{x}_j$, given by $\Delta = \frac{1}{2}|\mathbf{x}_i - \mathbf{x}_j| = \frac{1}{4}\|\mathbf{x}_i - \mathbf{x}_j\|_2^2$.

The SDM can also be implemented with continuous addresses, randomly placed on a unit hypersphere of $(N_{\text{in}} - 1)$ dimensions, denoted $\mathbb{S}^{N_{\text{in}}-1}$. The vector $\boldsymbol{\phi}_{\text{SDM}}(\mathbf{x})$ now indicates all addresses that lie within a hyperspherical cap centered at $\mathbf{x}$ with an angle $\arccos(b)$ between its central axis and the rim. The inner product $\boldsymbol{\phi}_{\text{SDM}}(\mathbf{x}_i)^\top \boldsymbol{\phi}_{\text{SDM}}(\mathbf{x}_j)$ is the number of addresses located in the overlapping area of two spherical caps centered at $\mathbf{x}_i$ and $\mathbf{x}_j$. While a calculation of this quantity, again, can be found in [8], it has not previously been connected to the kernel of an SDM. We simplify the previously published result and also derive a closed-form approximation, valid for highly sparse $\boldsymbol{\phi}_{\text{SDM}}$ (see Appendix B for details). The results are summarized below.

Property 3.2 (Kernel of an infinite SDM on the hypersphere)**.** *In the limit $N_\phi \to \infty$, the kernel of an SDM with N_ϕ memory slots, whose addresses are randomly drawn from $\mathbb{S}^{N_{\text{in}}-1}$, is given by*

$$K_{\text{SDM}}(\mathbf{x}_i, \mathbf{x}_j) = \frac{N_{\text{in}} - 2}{2\pi} \int_{\alpha_x}^{\alpha_b} \sin(\varphi)^{N_{\text{in}}-2} B\left[1 - \frac{\tan^2(\alpha_x)}{\tan^2(\varphi)}; \frac{N_{\text{in}} - 2}{2}, \frac{1}{2}\right] \mathrm{d}\varphi \tag{17}$$

where $\alpha_x = \frac{1}{2}\arccos(\mathbf{x}_i^\top \mathbf{x}_j)$, $\alpha_b = \arccos(b)$, and B is the incomplete Beta function. In the highly sparse regime, when $0.9 \lesssim b < 1$ and $\frac{1}{N_\phi}\|\boldsymbol{\phi}_{\text{SDM}}\|_0 \ll 1$, the kernel can be approximated with

$$K_{\text{SDM}}(\mathbf{x}_i, \mathbf{x}_j) \approx \frac{\hat{b}^{N_{\text{in}}-1}}{2\pi} B\left[1 - \left(\frac{\Delta}{\hat{b}}\right)^2; \frac{N_{\text{in}}}{2}, \frac{1}{2}\right] \tag{18}$$

where $\Delta = \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|_2$ and $\hat{b} = \sin(\arccos(b))$.

In conclusion, an infinitely large SDM with sparse internal representations $\boldsymbol{\phi}_{\text{SDM}}$, can be represented as a suboptimal case of a feed-forward SVM network with a translation-invariant kernel.

3.4 The modern Hopfield network is a recurrent SVM network

The Hopfield network [25] is, arguably, the most well-known model of auto-associative memory. In its modern form [34], it is a recurrent network of N neurons with the state $\mathbf{s}^{(t)}$, whose dynamics are governed by the energy and state update rule

$$E = -\sum_{\mu}^{M} F\left(\sum_{i}^{N} \xi_i^\mu s_i^{(t)}\right), \quad s_i^{(t+1)} = \text{sgn}\left[\sum_{\mu}^{M} \xi_i^\mu F'\left(\sum_{j \neq i}^{N} \xi_j^\mu s_j^{(t)}\right)\right] \tag{19}$$

where F is a smooth function, typically a sigmoid, polynomial, or exponential. This "generalized" Hopfield model has a long history [see, e.g., 1, 20, 24, 36] but has received renewed attention in recent years under the name *modern Hopfield network* (MHN) or *dense associative memory* [16, 34]. By comparing Eq. 19 with Eq. 12, the state update of the MHN can be identified as a special case of a suboptimal recurrent SVM network in the kernel form, with $k = F'$, $\mathbf{A} = \mathbf{1}$, and $\boldsymbol{\theta} = \mathbf{0}$ (since $f = 0.5$). With a linear $F'(x) = x$, the MHN reduces to the classical Hopfield network, which is a special case of the recurrent SVM network with the linear kernel $k(\mathbf{x}_i^\top \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j$.

Capacity. The storage capacity of the MHN has been shown to depend on the shape of F'. In the linear case, the capacity is famously limited to $\sim 0.1N$ patterns, depending on the precision of retrieval [2, 41]. If, on the other hand, F' is polynomial with degree p, the capacity scales as $M_{\max} \sim \mathcal{O}(N^p)$ [34], while an exponential F' endows the network with a capacity $M_{\max} \sim \mathcal{O}(e^N)$ [16]. From the perspective of the kernel memory framework, this scaling directly follows from the analysis in Sec. 3.2 with $k = F'$. In fact, in the regime of low errors, the kernel memory framework can also be used to derive a more precise capacity scaling for the classical Hopfield network. We first note that any one-shot learning rule that implies $\mathbf{A} > 0$ is equivalent to an SVM network where every stored pattern is a support vector. Such a heuristic is only likely to be close to the optimal solution and perform well in large networks with very few patterns, as high-dimensional linear SVMs trained on few patterns are highly likely to find solutions where all patterns are support vectors; this effect has been termed *support vector proliferation* [4]. Restricting the network to this regime limits the capacity to $M_{\max} \sim \mathcal{O}(\frac{N}{2\log N})$, consistent with the result in [41] (see Appendix A.2).

Iterative learning rules. Recently, Tyulmankov et al. [62] proposed a biologically plausible online learning algorithm for training an MHN. Depending on the implementation, it produces a storage capacity that scales as either $\sim 0.16N$ or $\sim N$. Commenting on this result, the authors noted that they are unaware of a learning algorithm capable of reaching the theoretical maximum capacity $\sim 2N$. Here, the perspective of kernel memory networks can be particularly helpful, as many of the algorithms that have been developed over the past two decades to optimize SVMs can be utilized for MHNs as well. For example, a network formulated in the feature form can be trained with the stochastic batch perceptron rule [13, 33], the passive aggressive rules [15], the minnorm rule [5], as well as with likelihood maximization applied to logistic regression [28, 45, 57]. In the kernel form, two of the most well-known online algorithms for training linear and non-linear SVMs are the Adatron [3] and the Kernel-Adatron [18]. A performance comparison between iterative learning and the modern Hopfield learning rule can be found in Appendix C.

Generalization. Viewing the MHN as a recurrent network of SVMs can also facilitate a more intuitive understanding of its ability to generalize, when used as a conventional classifier. In this setting, one designates a subset of the neurons as input units, and the remaining neurons as outputs. Given a set of input-output associations, one optimizes the memory patterns $\boldsymbol{\xi}^\mu$ using, for example, gradient descent. Such an experiment was performed by Krotov and Hopfield [34] on the MNIST data set, using a polynomial non-linearity $F(x) = x^p$. Results showed that the test error first improved as p increased from 2 to 3, but later deteriorated for high degrees, like $p = 20$. While it may be difficult to explain this behavior within an energy-based framework, it is entirely expected when viewed from the SVM perspective: a kernel of low polynomial degree has too few degrees of freedom to fit the classification boundary in the training set, causing *underfitting*, while a polynomial of too high degree grants the model too much flexibility, which results in *overfitting*.

The pseudoinverse learning rule. The coefficients in $\mathbf{A}$ are, in general, computed numerically, and cannot be written in closed form. However, in the special case when Eq. 8 is underdetermined, meaning $M < N_\phi$, a closed-form (but suboptimal) solution can be obtained using the *least-squares SVM* method [58]. The result is a generalized form of the *pseudoinverse learning rule* [48]. See Appendix D for details.

4 Kernel memory networks for continuous patterns

4.1 *Auto*-associative memory as a recurrent interpolation network

So far, we have considered memory models designed to store only bipolar patterns. We now relax this constraint and allow patterns to be continuous-valued. We first observe that any set of patterns $\mathbf{X} \in \mathbb{R}^{N \times M}$ can be made fixed points of the dynamics by training each neuron i to interpolate ξ_i^μ when the rest of the network is initialized in $\boldsymbol{\xi}^\mu$, for every pattern μ. Assuming that the model is equipped with a kernel that allows for each fixed point to also be attracting, we can ensure that a lower bounding estimate of the size of the attractor basin is maximized by finding the interpolation with minimum weight norm (see Appendix E.1 for proof). These results are summarized below.

Property 4 (Robust auto-associative memory with continuous patterns)**.** *Suppose that the dynamics of a recurrent auto-associative memory network evolve according to the synchronous update rule*

$$\mathbf{s}^{(t+1)} = \mathbf{X}\mathbf{K}^\dagger K(\mathbf{X}, \mathbf{s}^{(t)}) \tag{20}$$

where $\mathbf{K} = K(\mathbf{X}, \mathbf{X}) = \boldsymbol{\phi}(\mathbf{X})^\top \boldsymbol{\phi}(\mathbf{X})$ *is the kernel matrix and* $\mathbf{K}^\dagger$ *its Moore-Penrose pseudoinverse, where* $\mathbf{K}^\dagger = \mathbf{K}^{-1}$ *if* $\boldsymbol{\phi}(\mathbf{X})$ *is full column rank. Then, the dynamics of the network is guaranteed to have the fixed points* $\mathbf{X}$*. Moreover, if the points are attracting, Eq. 20 maximizes a lower bound of the attractor basin sizes.*

4.2 A recurrent interpolation network with exponential capacity

Memory models for continuous data have generally received less attention than their binary counterparts [26, 32]. Recently, however, Ramsauer et al. [55] proposed an energy-based model capable of storing an exponential number of continuous-valued patterns (we will refer to this model as the softmax network). While the structure of this model is similar to Eq. 20, it cannot be analyzed within

the framework of Property 4, as it involves a kernel that is neither symmetric nor positive-definite [66].

Nonetheless, we will in this section demonstrate that it is possible to use conventional kernel methods to design an attractor network with exponential capacity for continuous patterns. We utilize the properties of the SDM by using a translation-invariant kernel with a fixed spatial scale r. For the sake of simplicity, we choose the exponential power kernel (Exp_β)

$$K_{\text{exp}_\beta}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\left(\frac{1}{r}\|\mathbf{x}_i - \mathbf{x}_j\|_2\right)^\beta\right] \tag{21}$$

where $\beta, r > 0$. These parameters determine the shape of the attractor basin that surrounds each pattern. While r roughly sets the radius of attraction, β represents an inverse temperature which changes the steepness of the boundary of the attractor basin. Moreover, as long as the patterns are unique, the kernel matrix is invertible and we have $\mathbf{K}^\dagger_{\text{exp}_\beta} = \mathbf{K}^{-1}_{\text{exp}_\beta}$ [43].

We will now analyze the noise robustness and storage capacity of this model. To make the analysis tractable, we will operate in the regime of low temperatures, meaning the limit $\beta \to \infty$. We first establish the following three properties.

Property 5.1 (The Exp_β network at zero temperature)**.** *Given a set of unique patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^M$ with $\min_{\mu,\nu\neq\mu}\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2 > r$, the state update rule for the Exp_β network at $\beta \to \infty$ reduces to*

$$\mathbf{s}^{(t+1)} = \mathbf{X}\,\Theta(r - \|\mathbf{X} - \mathbf{s}^{(t)}\|_2) \tag{22}$$

where $\Theta(\cdot)$ is the Heaviside function with $\Theta(0) = e^{-1}$ (see Appendix E.2.1).

Remark. In geometrical terms, Property 5.1 states that the boundary of the basin of attraction surrounding each pattern becomes a sharp $(N-1)$-dimensional hypersphere with radius r in the limit $\beta \to \infty$. For lower, finite β, the spherical boundary becomes increasingly fuzzy. From the perspective of an energy landscape, each pattern lies in an N-dimensional energy minimum with infinitely steep walls when $\beta \to \infty$. As β is lowered, the barriers become progressively smoother.

Property 5.2 (Convergence in one step)**.** *Given a set of unique patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^M$ with $\min_{\mu,\nu\neq\mu}\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2 > 2r$, the Exp_β network at $\beta \to \infty$, initialized at $\mathbf{s}^{(0)} = \boldsymbol{\xi}^\mu + \Delta\boldsymbol{\xi}$, will converge to $\boldsymbol{\xi}^\mu$ in one step if $\|\Delta\boldsymbol{\xi}\|_2 < r$.*

Property 5.3 (No spurious attractors)**.** *Given a set of unique patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^M$ with $\min_{\mu,\nu\neq\mu}\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2 > 2r$ and $\nexists\mu : \|\boldsymbol{\xi}^\mu\|_2 = r/(1-e^{-1})$, the only attractors of the dynamics of the Exp_β network at $\beta \to \infty$ are the points $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^M$, together with $\mathbf{0}$ if $\nexists\mu : \|\boldsymbol{\xi}^\mu\|_2 \leq r$.*

Remark. Properties 5.2 and 5.3 can be shown to be true simply by inserting the expression $\mathbf{s}^{(0)} = \boldsymbol{\xi}^\mu + \Delta\boldsymbol{\xi}$ in Eq. 22. Assuming no overlaps between the basins of attraction, a quick calculation shows that $\mathbf{s}^{(1)} = \boldsymbol{\xi}^\mu$ if $\|\Delta\boldsymbol{\xi}\|_2 < r$. If, on the other hand, the network is initialized such that $\|\mathbf{s}^{(0)} - \boldsymbol{\xi}^\mu\|_2 > r, \forall\mu$, one always obtains $\mathbf{s}^{(2)} = \boldsymbol{\xi}^0$, where $\boldsymbol{\xi}^0$ is either $\mathbf{0}$ or the pattern closest to $\mathbf{0}$. In other words, the network recalls a pattern only if the initialization is close enough to it. If located far from *all* patterns, the network assumes an "agnostic" state, represented either by the origin or the pattern closest to the origin (if the origin happens to be located within a basin of attraction).

In the following two properties, we evaluate how the radius of attraction r determines the maximum input noise tolerance and storage capacity.

Property 6 (Robustness to white noise)**.** *Assume that we are given a set of unique patterns $\boldsymbol{\xi}^1, \ldots, \boldsymbol{\xi}^M \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$ with $\min_{\mu,\nu\neq\mu}\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2 > 2r$, and that the Exp_β network is initialized in a distorted pattern $\mathbf{s}^{(0)} = \boldsymbol{\xi}^\mu + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_N)$. Then, at $\beta \to \infty$, the maximum noise variance $\sigma^2_{\max}$ with which $\boldsymbol{\xi}^\mu$ can be recovered in at least 50% of trials is*

$$\sigma^2_{\max} = r^2/N\ . \tag{23}$$

Property 7 (Exponential storage capacity)**.** *At $\beta \to \infty$, and for $N \gg 1$, the average maximum number of patterns sampled from $\mathcal{N}(\mathbf{0}, \mathbf{I}_N)$ that the Exp_β network can store and recall without errors is lower-bounded according to*

$$M_{\max} \geq \sqrt{2\sqrt{\pi N}(1 - 2\sigma^2_{\max})}\exp\left[\frac{N(1-2\sigma^2_{\max})^2}{8}\right] \tag{24}$$

where $\sigma^2_{\max}$ is the maximum white noise variance tolerated by the network.

Remark. Proofs can be found in Appendices E.2.2 and E.2.3. Note that Property 7 is valid in the range $\sigma^2_{\max} \lesssim 1/2$. While the bounds are fairly tight at the upper end of the range, they become loose when $\sigma^2_{\max} \to 0$. In this limit, which is equivalent to $r \to 0$, the storage capacity tends to infinity, as the risk of interference between patterns vanishes when their radius of attraction becomes infinitesimal.

Comparison to the softmax network. If patterns are randomly placed on a hypersphere instead of being normally distributed, the state update rule in Eq. 22 reduces to the form $\mathbf{s}^{(t+1)} = \mathbf{X}\,\Theta(\mathbf{X}^\top \mathbf{s}^{(t)} - \theta)$, where θ is a fixed threshold. While the capacity remains exponential (see Appendix E.3.1), the basin of attraction surrounding each pattern now forms a spherical cap instead of a ball. We can compare this to the softmax network at zero temperature, given by $\mathbf{s}^{(t+1)} = \lim_{\beta\to\infty} \mathbf{X}\,\mathrm{softmax}(\beta \mathbf{X}^\top \mathbf{s}^{(t)}) = \mathbf{X}\arg\max(\mathbf{X}^\top \mathbf{s}^{(t)})$. This model differs from the Exp_β only in a replacement of Θ with $\arg\max$. This changes the shape of the attractor basins from spherical caps to Voronoi cells, which parcellate the entire surface of the hypersphere into a Voronoi diagram (see Fig. 2). The boundary of each basin is now no longer radially symmetric around a pattern, but instead extends as far as possible in all directions. Consequently, at $\beta \to \infty$, the softmax network has larger attractor basins and always converges to one of the stored patterns, regardless of the initialization point (assuming this is not precisely on a boundary). In contrast, the Exp_β network may converge to the origin if initialized far from all patterns. This can be interpreted as an agnostic response, which indicates that the model cannot associate the input query with any of its stored patterns.

5 Discussion

Biological interpretation. Kernel memory networks can be mapped to the anatomical properties of biological neurons. Consider an individual neuron in the feature form of the recurrent network (Eq. 9). The state of neighboring neurons $\mathbf{s}$ is first transformed through $\boldsymbol{\phi}(\mathbf{s})$ and thereafter projected to the neuron through the weight matrix $(\mathbf{A} \odot \mathbf{X})\boldsymbol{\phi}(\mathbf{X})^\top$. When the kernel is polynomial of degree p, so that $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j + 1)^p$, the transformation $\boldsymbol{\phi}(\mathbf{s})$ consists of all elements in $\mathbf{s}$ and their cross-terms, up to degree p. The input to each neuron, in other words, consists of the states of all other neurons, as well as all possible combinations of their multiplicative interactions. This neuron model can be viewed as a generalized form of, for example, the multiconnected neuron [47], the clusteron [42], or the sigma-pi unit [56, p. 73]. These are all perceptrons that include multiplicative input interactions as a means to model synaptic cross-talk and cluster-sensitivity on non-linear dendrites [53] (see Fig. 1).

In the kernel form (Eq. 10), each neuron is, again, implicitly comprised of a two-stage process, whereby the raw input $\mathbf{s}$ is first transformed through the function $K(\mathbf{X}, \mathbf{s})$ and then projected through the weight matrix $\mathbf{A} \odot \mathbf{X}$. For any inner-product kernel $K = k(\mathbf{x}_i^\top \mathbf{x}_j)$, this representation can be directly identified as a two-layer neural network, where the hidden layer is defined by the weights $\mathbf{X}$ and the activation function k. This interpretation of the recurrent network was recently proposed in [34, 35] and discussed in relation to hippocampal-cortical interactions involved in memory storage and recall; it is particularly reminiscent of the hippocampal indexing theory [6, 59]. However, the kernel form can also be viewed as a network in which each *individual* neuron is a generalized form of the two-layered pyramidal cell model [51, 52]. This was originally proposed as an abstract neuron model augmented with non-linear dendritic processing [40]. It should be noted, however, that the idea of interpreting kernel methods as neural networks has a longer history, and has been extensively analyzed in the case of, for example, radial basis functions [49, 50]. For further details, see Appendix F.

Summary. We have shown that conventional kernel methods can be used to derive the weights for hetero- and auto-associative memory networks storing binary or continuous-valued patterns with maximal noise tolerance. The result is a family of optimal memory models, which we call *kernel memory networks*, which includes the SDM and MHN as special cases. This unifying framework facilitates an intuitive understanding of the storage capacity of memory models and offers new ways to biologically interpret these in terms of non-linear dendritic integration. This work formalizes the links between kernel methods, attractor networks, and models of dendritic processing.

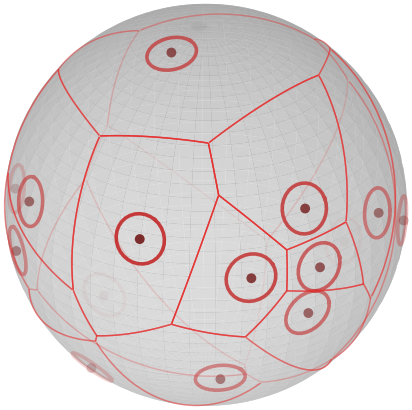

Figure 2: Plot of random patterns on $\mathbb{S}^2$ together with attractor basins at $\beta \to \infty$. Dots represent patterns ($M = 17$) while thick and thin red lines correspond to the boundaries of the attractor basins according to the Exp_β network and the softmax network, respectively. The radius of the circular boundaries has been set to half the minimum pairwise distance between the patterns.

Future work. A unifying theoretical framework for memory modeling can be useful for the development of both improved bio-plausible memory models and for machine learning applications. First, recognizing that there exists algorithms for training optimally noise-robust classifiers and adapting these to biological constraints can aid in the development of normative synaptic three-factor learning rules [22]. Second, the theoretical link between neuron models, kernel functions, and storage capacity enables one to fit kernel memory networks to neurophysiological data and to analyze the computational properties of biophysically informed memory models. Finally, our unifying framework reveals that most memory models differ only in the choice of kernel (model complexity) and Lagrange parameters (model precision). This categorization simplifies the tailoring of memory models to their application, and allows for the design of models whose properties fundamentally can depart from kernel memory networks, by, for example, choosing kernels not associated with a reproducing kernel Hilbert space.

Acknowledgments and Disclosure of Funding

This study was supported by funding from the Swiss government's ETH Board of the Swiss Federal Institutes of Technology to the Blue Brain Project, a research center of the École Polytechnique Fédérale de Lausanne (EPFL).

References

[1] Abbott, L. F. and Arian, Y. (1987). "Storage capacity of generalized networks". *Physical Review A* 36, pp. 5091–5094.

[2] Amit, D. J., Gutfreund, H., and Sompolinsky, H. (1985). "Storing infinite numbers of patterns in a spin-glass model of neural networks". *Physical Review Letters* 55, pp. 1530–1533.

[3] Anlauf, J. K. and Biehl, M. (1989). "The AdaTron: an adaptive perceptron algorithm". *Europhysics Letters* 10, pp. 687–692.

[4] Ardeshir, N., Sanford, C., and Hsu, D. (2021). "Support vector machines and linear regression coincide with very high-dimensional features". *Advances in Neural Information Processing Systems 35*.

[5] Bansal, Y., Advani, M., Cox, D. D., and Saxe, A. M. (2018). "Minnorm training: an algorithm for training over-parameterized deep neural networks". Preprint. arXiv: 1806.00730.

[6] Barry, D. N. and Maguire, E. A. (2019). "Remote memory and the hippocampus: a constructive critique". *Trends in Cognitive Sciences* 23, pp. 128–142.

[7] Boros, G. and Moll, V. (2004). *Irresistible integrals: symbolics, analysis and experiments in the evaluation of integrals*. Cambridge University Press.

[8] Bricken, T. and Pehlevan, C. (2021). "Attention approximates sparse distributed memory". *Advances in Neural Information Processing Systems 34*.

[9] Chen, L. and Xu, S. (2020). "Deep neural tangent kernel and Laplace kernel have the same RKHS". *Proceedings of the 2021 International Conference on Learning Representations*.

[10] Cho, Y. and Saul, L. (2009). "Kernel methods for deep learning". *Advances in Neural Information Processing Systems 22*.

[11] Chou, P. A. (1989). "The capacity of the Kanerva associative memory". *IEEE Transactions on Information Theory* 35, pp. 281–298.

[12] Cortes, C. and Vapnik, V. (1995). "Support-vector networks". *Machine Learning* 20, pp. 273–297.

[13] Cotter, A., Shalev-Shwartz, S., and Srebro, N. (2012). "The kernelized stochastic batch perceptron". *Proceedings of the 29th International Conference on Machine Learning*, pp. 739–746.

[14] Cover, T. M. (1965). "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition". *IEEE Transactions on Electronic Computers* EC-14, pp. 326–334.

[15] Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). "Online passive-aggressive algorithms". *Journal of Machine Learning Research* 7, pp. 551–585.

[16] Demircigil, M., Heusel, J., Löwe, M., Upgang, S., and Vermet, F. (2017). "On a model of associative memory with huge storage capacity". *Journal of Statistical Physics* 168, pp. 288–299.

[17] Forrest, B. M. (1988). "Content-addressability and learning in neural networks". *Journal of Physics A: Mathematical and General* 21, pp. 245–255.

[18] Frieß, T.-T., Cristianini, N., and Campbell, C. (1998). "The Kernel-Adatron algorithm: a fast and simple learning procedure for support vector machines". *Proceedings of the 15th International Conference on Machine Learning*, pp. 188–196.

[19] Gardner, E. (1987a). "Maximum storage capacity in neural networks". *Europhysics Letters* 4, pp. 481–485.

[20] Gardner, E. (1987b). "Multiconnected neural network models". *Journal of Physics A: Mathematical and General* 20, pp. 3453–3464.

[21] Gardner, E. (1988). "The space of interactions in neural network models". *Journal of Physics A: Mathematical and General* 21, pp. 257–270.

[22] Gerstner, W., Lehmann, M., Liakoni, V., Corneil, D., and Brea, J. (2018). "Eligibility traces and plasticity on behavioral time scales: experimental support of neohebbian three-factor learning rules". *Frontiers in Neural Circuits* 12.

[23] Graves, A., Wayne, G., and Danihelka, I. (2014). "Neural Turing machines". Preprint. arXiv:1410.5401.

[24] Hintzman, D. L. (1984). "MINERVA 2: A simulation model of human memory". *Behavior Research Methods, Instruments, & Computers* 16, pp. 96–101.

[25] Hopfield, J. J. (1982). "Neural networks and physical systems with emergent collective computational abilities". *Proceedings of the National Academy of Sciences* 79, pp. 2554–2558.

[26] Hopfield, J. J. (1984). "Neurons with graded response have collective computational properties like those of two-state neurons". *Proceedings of the National Academy of Sciences* 81, pp. 3088–3092.

[27] Jacot, A., Gabriel, F., and Hongler, C. (2018). "Neural tangent kernel: convergence and generalization in neural networks". *Advances in Neural Information Processing Systems 31*.

[28] Ji, Z., Srebro, N., and Telgarsky, M. (2021). "Fast margin maximization via dual acceleration". *Proceedings of the 38th International Conference on Machine Learning*, pp. 4860–4869.

[29] Kanerva, P. (1988). *Sparse distributed memory*. MIT Press.

[30] Keeler, J. D. (1988). "Comparison between Kanerva's SDM and Hopfield-type neural networks". *Cognitive Science* 12, pp. 299–329.

[31] Kepler, T. B. and Abbott, L. F. (1988). "Domains of attraction in neural networks". *Journal de Physique* 49, pp. 1657–1662.

[32] Koiran, P. (1994). "Dynamics of discrete time, continuous state Hopfield networks". *Neural Computation* 6, pp. 459–468.

[33] Krauth, W. and Mezard, M. (1987). "Learning algorithms with optimal stability in neural networks". *Journal of Physics A: Mathematical and General* 20, pp. L745–L752.

[34] Krotov, D. and Hopfield, J. J. (2016). "Dense associative memory for pattern recognition". *Advances in Neural Information Processing Systems 29*, pp. 1172–1180.

[35] Krotov, D. and Hopfield, J. J. (2020). "Large associative memory problem in neurobiology and machine learning". *Proceedings of the 9th International Conference on Learning Representations*.

[36] Lee, Y. C., Doolen, G., Chen, H. H., Sun, G. Z., Maxwell, T., Lee, H. Y., and Giles, C. L. (1986). "Machine learning using a higher order correlation network". *Physica D: Nonlinear Phenomena* 22, pp. 276–306.

[37] Lee, Y. and Kim, W. C. (2014). *Concise formulas for the surface area of the intersection of two hyperspherical caps*. Technical Report. Department of Industrial and Systems Engineering, KAIST.

[38] Li, S. (2011). "Concise formulas for the area and volume of a hyperspherical cap". *Asian Journal of Mathematics and Statistics* 4, pp. 66–70.

[39] MacKay, D. J. C. (2003). *Information theory, inference and learning algorithms*. Cambridge University Press.

[40] Major, G., Larkum, M. E., and Schiller, J. (2013). "Active properties of neocortical pyramidal neuron dendrites". *Annual Review of Neuroscience* 36, pp. 1–24.

[41] McEliece, R., Posner, E., Rodemich, E., and Venkatesh, S. (1987). "The capacity of the Hopfield associative memory". *IEEE Transactions on Information Theory* 33, pp. 461–482.

[42] Mel, B. W. (1991). "The clusteron: toward a simple abstraction for a complex neuron". *Advances in Neural Information Processing Systems 4*, pp. 35–42.

[43] Micchelli, C. A. (1986). "Interpolation of scattered data: distance matrices and conditionally positive definite functions". *Constructive Approximation* 2, pp. 11–22.

[44] Millidge, B., Salvatori, T., Song, Y., Lukasiewicz, T., and Bogacz, R. (2022). "Universal Hopfield networks: a general framework for single-shot associative memory models". *Proceedings of the 39th International Conference on Machine Learning*, pp. 15561–15583.

[45] Nacson, M. S., Lee, J., Gunasekar, S., Savarese, P. H. P., Srebro, N., and Soudry, D. (2019). "Convergence of gradient descent on separable data". *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3420–3428.

[46] Neal, R. M. (1996). "Priors for infinite networks". *Bayesian Learning for Neural Networks*. Ed. by Neal, R. M. Vol. 118. Lecture Notes in Statistics. Springer, pp. 29–53.

[47] Peretto, P. and Niez, J. J. (1986). "Long term memory storage capacity of multiconnected neural networks". *Biological Cybernetics* 54, pp. 53–63.

[48] Personnaz, L., Guyon, I., and Dreyfus, G. (1986). "Collective computational properties of neural networks: New learning mechanisms". *Physical Review A* 34, pp. 4217–4228.

[49] Poggio, T. and Girosi, F. (1990a). "Networks for approximation and learning". *Proceedings of the IEEE* 78, pp. 1481–1497.

[50] Poggio, T. and Girosi, F. (1990b). "Regularization algorithms for learning that are equivalent to multilayer networks". *Science* 247, pp. 978–982.

[51] Poirazi, P., Brannon, T., and Mel, B. W. (2003). "Pyramidal neuron as two-layer neural network". *Neuron* 37, pp. 989–999.

[52] Poirazi, P. and Mel, B. W. (2001). "Impact of active dendrites and structural plasticity on the memory capacity of neural tissue". *Neuron* 29, pp. 779–796.

[53] Polsky, A., Mel, B. W., and Schiller, J. (2004). "Computational subunits in thin dendrites of pyramidal cells". *Nature Neuroscience* 7, pp. 621–627.

[54] Radhakrishnan, A., Belkin, M., and Uhler, C. (2020). "Overparameterized neural networks implement associative memory". *Proceedings of the National Academy of Sciences* 117, pp. 27162–27170.

[55] Ramsauer, H., Schäfl, B., Lehner, J., Seidl, P., Widrich, M., Gruber, L., Holzleitner, M., Adler, T., Kreil, D., Kopp, M. K., Klambauer, G., Brandstetter, J., and Hochreiter, S. (2021). "Hopfield networks is all you need". *Proceedings of the 9th International Conference on Learning Representations*.

[56] Rumelhart, D. E. and McClelland, J. L. (1986). *Parallel distributed processing: explorations in the microstructure of cognition*. Vol. 1. MIT Press.

[57] Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. (2018). "The implicit bias of gradient descent on separable data". *The Journal of Machine Learning Research* 19, pp. 2822–2878.

[58] Suykens, J. and Vandewalle, J. (1999). "Least squares support vector machine classifiers". *Neural Processing Letters* 9, pp. 293–300.

[59] Teyler, T. J. and Rudy, J. W. (2007). "The hippocampal indexing theory and episodic memory: updating the index". *Hippocampus* 17, pp. 1158–1169.

[60] Tony Cai, T. and Jiang, T. (2012). "Phase transition in limiting distributions of coherence of high-dimensional random matrices". *Journal of Multivariate Analysis* 107, pp. 24–39.

[61] Tsai, Y.-H. H., Bai, S., Yamada, M., Morency, L.-P., and Salakhutdinov, R. (2019). "Transformer dissection: an unified understanding for transformer's attention via the lens of kernel". *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4344–4353.

[62] Tyulmankov, D., Fang, C., Vadaparty, A., and Yang, G. R. (2021). "Biological learning in key-value memory networks". *Advances in Neural Information Processing Systems 34*.

[63] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). "Attention is all you need". *Advances in Neural Information Processing Systems 30*.

[64] Weston, J., Chopra, S., and Bordes, A. (2015). "Memory networks". *Proceedings of the 2015 International Conference on Learning Representations*.

[65] Williams, C. (1996). "Computing with infinite networks". *Advances in Neural Information Processing Systems 9*.

[66] Wright, M. A. and Gonzalez, J. E. (2021). "Transformers are deep infinite-dimensional non-Mercer binary kernel machines". Preprint. arXiv: 2106.01506.

[67] Wu, Y., Wayne, G., Graves, A., and Lillicrap, T. (2018). "The Kanerva machine: a generative distributed memory". *Proceedings of the 2018 International Conference on Learning Representations*.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] *Our work is entirely theoretical and the limitations of our approach is treated in the main text when discussing, for example, the validity and assumptions of our derivations.*
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [N/A]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] *This is included in all theoretical results, or in the explanatory text preceding them.*
   (b) Did you include complete proofs of all theoretical results? [Yes] *These can be found in the Appendix. Pointers to the correct sections are given in the main text.*
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendices

A Derivation of storage capacity scaling

A.1 Optimal storage: The scaling of effective input dimensionality

Suppose that the kernel is a homogeneous polynomial of degree $p \ll N$, meaning $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^p$. This implies that the associated feature map $\boldsymbol{\phi}(\mathbf{x})$ contains all monomials of degree p composed of the entries in $\mathbf{x}$. Moreover, given that each entry x_i is ± 1, each monomial in $\boldsymbol{\phi}$ can be written as an interaction term of the form $x_1^{p_1} x_2^{p_2} \cdots x_N^{p_N}$, where $p_i \in \{0, 1\}$ and $\sum_i^N p_i \leq p$ (i.e., no factor x_i has an exponent higher than 1 and the sum of exponents is $\leq p$). The reason for this is that

$$x_i^{n_i} = \begin{cases} 1\,, & \text{if } n_i \text{ even} \\ x_i\,, & \text{if } n_i \text{ odd}\,. \end{cases} \tag{25}$$

The number of unique interaction terms of precisely degree p (the highest degree) is $\binom{N}{p}$. As the binomial coefficient is known to be bounded according to

$$\left(\frac{N}{p}\right)^p \leq \binom{N}{p} \leq \left(\frac{Ne}{p}\right)^p \tag{26}$$

we obtain $\binom{N}{p} \sim \mathcal{O}(N^p)$, for p fixed. Thus, the effective dimensionality of $\boldsymbol{\phi}$ scales like $\mathcal{O}(N^p)$.

For the exponential kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(\mathbf{x}_i^\top \mathbf{x}_j) = \sum_{p=0}^{\infty} (\mathbf{x}_i^\top \mathbf{x}_j)^p / p!$, we first note that the monomials in $\boldsymbol{\phi}$ now will be interaction terms of all degrees $p = 0, \ldots, N$. No monomial of degree $p > N$ will be possible. The total number of unique interaction terms will therefore be

$$\sum_{p=0}^{N} \binom{N}{p} = 2^N = e^{N \log 2} \tag{27}$$

where the first equality can be found in [7, p. 14]. This gives us an effective dimensionality of $\boldsymbol{\phi}$ that scales like $\mathcal{O}(e^N)$.

A.2 One-shot storage: The scaling of the support vector proliferation regime

As shown recently in [4], support vector proliferation for an SVM trained on M random patterns drawn uniformly from $\{\pm 1\}^N$ occurs in the regime $N \gtrsim 2M \log M$. Solving for M gives us the scaling

$$M \lesssim \frac{N}{2W_0(\frac{N}{2})} \tag{28}$$

where W_0 is the principal branch of the Lambert function. The largest number of patterns that can be stored in this regime is thus

$$M_{\max} \approx \frac{N}{2W_0(\frac{N}{2})}\,. \tag{29}$$

Using the property $W_0(x) = \log x - \log\log x + o(1)$, we can write

$$W_0\left(\frac{N}{2}\right) \sim \mathcal{O}(\log N) \tag{30}$$

which yields

$$M_{\max} \sim \mathcal{O}\left(\frac{N}{2\log N}\right)\,. \tag{31}$$

B Kernel of an infinite SDM on the hypersphere

B.1 Derivation of Eq. 17.

We follow the same steps as [8], with additional simplifications towards the end. As stated in the main text, we seek to calculate the overlapping area of two hyperspherical caps. A formula for this can be found in [37], and can be written as

$$A_\cap = A_\triangledown(R, \alpha_{\min}, \alpha_2) + A_\triangledown(R, \alpha_v - \alpha_{\min}, \alpha_1) \tag{32}$$

where

$$A_\triangledown(R, \alpha_{\min}, \alpha_2) = \frac{\pi^{\frac{N_{\text{in}}-1}{2}}}{\Gamma(\frac{N_{\text{in}}-1}{2})} R^{N_{\text{in}}-1} \int_{\alpha_{\min}}^{\alpha_2} \sin(\varphi)^{N_{\text{in}}-2} I_{1-\frac{\tan^2(\alpha_{\min})}{\tan^2(\varphi)}}\left[\frac{N_{\text{in}}-2}{2}, \frac{1}{2}\right] \mathrm{d}\varphi \tag{33}$$

where R is the radius, I is the regularized incomplete Beta function, and

$$\alpha_1 = \alpha_2 = \arccos(b) \tag{34}$$

$$\alpha_v = \arccos(\mathbf{x}_i^\top \mathbf{x}_j) \tag{35}$$

$$\alpha_{\min} = \arctan\left(\frac{\cos(\alpha_1)}{\cos(\alpha_2)\sin(\alpha_v)} - \frac{1}{\tan(\alpha_v)}\right) \tag{36}$$

$$R = 1\ . \tag{37}$$

We insert Eq. 34 in 36 and obtain

$$\alpha_{\min} = \arctan\left(\frac{1}{\sin(\alpha_v)} - \frac{\cos(\alpha_v)}{\sin(\alpha_v)}\right) = \arctan\left(\tan\left(\frac{\alpha_v}{2}\right)\right) = \frac{\alpha_v}{2} \tag{38}$$

where we have used $\tan(\alpha/2) = (1-\cos(\alpha))/\sin(\alpha)$. Eq. 38 also follows from the symmetry of the problem. This result yields

$$A_\cap = A_\triangledown(R, \alpha_{\min}, \alpha_2) + A_\triangledown(R, \alpha_v - \alpha_{\min}, \alpha_1) = 2A_\triangledown(R, \alpha_{\min}, \alpha_1)\ . \tag{39}$$

We rewrite the regularized incomplete Beta function as

$$I_{1-\frac{\tan^2(\alpha_{\min})}{\tan^2(\varphi)}}\left[\frac{N_{\text{in}}-2}{2}, \frac{1}{2}\right] = \frac{\Gamma(\frac{N_{\text{in}}-1}{2})}{\Gamma(\frac{N_{\text{in}}-2}{2})\Gamma(\frac{1}{2})} B\left[1 - \frac{\tan^2(\alpha_{\min})}{\tan^2(\varphi)}; \frac{N_{\text{in}}-2}{2}, \frac{1}{2}\right] \tag{40}$$

where B is the incomplete Beta function. The area of an $(N_{\text{in}}-1)$-dimensional hypersphere is

$$A_\circ = \frac{2\pi^{\frac{N_{\text{in}}}{2}}}{\Gamma(\frac{N_{\text{in}}}{2})} R^{N_{\text{in}}-1}\ . \tag{41}$$

We insert Eq. 40 in 33 and use the result in Eq. 39. Using the notation $\alpha_b = \alpha_1$ and $\alpha_x = \alpha_{\min}$, and the identities

$$\frac{\Gamma(\frac{N_{\text{in}}}{2})}{\Gamma(\frac{N_{\text{in}}-2}{2})} = \frac{N_{\text{in}}-2}{2}, \quad \Gamma\left(\frac{1}{2}\right) = \sqrt{\pi}\ , \tag{42}$$

the ratio between the overlapping area of the hyperspherical caps and the complete are of the hypersphere can now be calculated as

$$\frac{A_\cap}{A_\circ} = \frac{N_{\text{in}}-2}{2\pi} \int_{\alpha_x}^{\alpha_b} \sin(\varphi)^{N_{\text{in}}-2} B\left[1 - \frac{\tan^2(\alpha_x)}{\tan^2(\varphi)}; \frac{N_{\text{in}}-2}{2}, \frac{1}{2}\right] \mathrm{d}\varphi\ . \tag{43}$$

B.2 Derivation of Eq. 18.

For a large bias $b \gtrsim 0.9$, which is equivalent to a small angle $\alpha_b = \arccos(b)$, the hyperspherical caps surrounding $\mathbf{x}_i$ and $\mathbf{x}_j$ will be very small in relation to the whole hypersphere. In this case, we can neglect the curvature of the hyperspherical surface and project the area of the hyperspherical cap to the plane that cuts through the rims of the cap. This projection is a $(N_{\text{in}}-1)$-dimensional hyperball

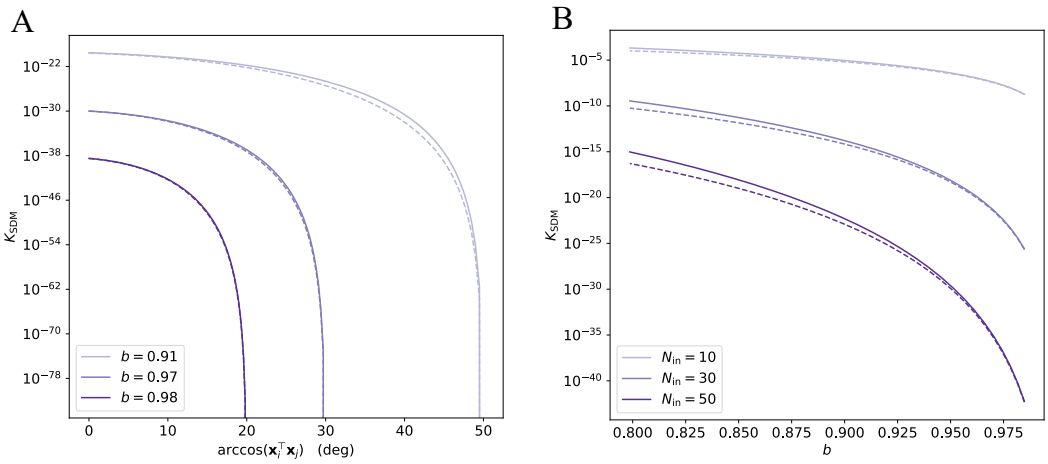

Figure B.1: Plot of the kernel of an infinite SDM on the hypersphere, $K_{\mathrm{SDM}}(\mathbf{x}_i, \mathbf{x}_j)$, as a function of (A) the angle between $\mathbf{x}_i$ and $\mathbf{x}_j$, and (B) the bias b. Solid lines represent the exact solution in Eq. 43, and dashed lines the approximation in Eq. 48. Parameter values: (A) $N_{\mathrm{in}} = 50$; (B) $\arccos(\mathbf{x}_i^\top \mathbf{x}_j) = \arccos(b)$.

(we will refer to it as a mini-ball). In three dimensions, for example, the projection of a spherical cap to the plane constitutes a disk, which is a 2-dimensional ball. The radius of each mini-ball is

$$\hat{b} = \sin(\arccos(b)) \tag{44}$$

and the half-distance between the centers of the mini-balls is

$$\Delta = \frac{1}{2}\|\mathbf{x}_i - \mathbf{x}_j\|_2 \ . \tag{45}$$

We estimate the overlapping area of the hyperspherical caps by calculating the overlapping volume of the mini-balls in $(N_{\mathrm{in}} - 1)$ dimensions. A formula for the overlapping volume of two hyperballs can be found in [38] and is

$$V_\cap = \frac{\pi^{\frac{N_{\mathrm{in}}-1}{2}}}{\Gamma(\frac{N_{\mathrm{in}}+1}{2})} \hat{b}^{N_{\mathrm{in}}-1} I_{1-\left(\frac{\Delta}{\hat{b}}\right)^2}\left[\frac{N_{\mathrm{in}}}{2}, \frac{1}{2}\right] \ . \tag{46}$$

We rewrite the regularized incomplete Beta function as

$$I_{1-\left(\frac{\Delta}{2\hat{b}}\right)^2}\left[\frac{N_{\mathrm{in}}}{2}, \frac{1}{2}\right] = \frac{\Gamma(\frac{N_{\mathrm{in}}+1}{2})}{\Gamma(\frac{N_{\mathrm{in}}}{2})\Gamma(\frac{1}{2})} B\left[1 - \left(\frac{\Delta}{\hat{b}}\right)^2; \frac{N_{\mathrm{in}}}{2}, \frac{1}{2}\right] \tag{47}$$

and insert Eq. 47 in 46. The ratio between the overlapping area of the hyperspherical caps and the complete area of the hypersphere can now be estimated as

$$\frac{A_\cap}{A_\circ} \approx \frac{V_\cap}{A_\circ} = \frac{\hat{b}^{N_{\mathrm{in}}-1}}{2\pi} B\left[1 - \left(\frac{\Delta}{\hat{b}}\right)^2; \frac{N_{\mathrm{in}}}{2}, \frac{1}{2}\right] \ . \tag{48}$$

A comparison of the exact solution in Eq. 43 and the approximation in Eq. 48 can be seen in Fig. B.1.

C Iterative learning in an SVM network

We will in this section compare the noise tolerance of a single neuron in an SVM network when trained with an iterative learning rule, and when configured according to the MHN. First, we choose to equip the neuron with the feature map $\boldsymbol{\phi}_{\mathrm{pairs}}(\mathbf{x})$, which consists of all unique pairs of cross-terms $x_i x_j$, $i \neq j$. This yields a storage capacity scaling of $\mathcal{O}(N^2)$, and we therefore parameterize the storage load as M/N^2. We train the weights of the neuron either with the stochastic batch perceptron

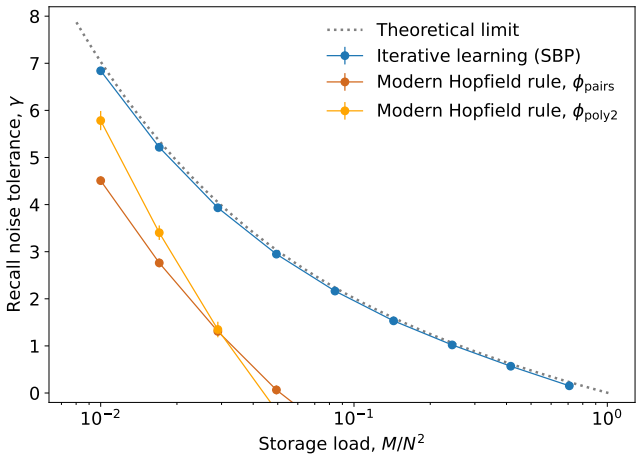

Figure C.1: Plot of the noise tolerance γ (mean $\pm$ s.e.m. over 20 simulations) as a function of the storage load for a single SVM neuron with $N = 10^2$ inputs, trained with the stochastic batch perceptron (SBP) and the modern Hopfield rule. The SBP uses the feature map $\boldsymbol{\phi}_{\text{pairs}}$, while the modern Hopfield rule is applied to both $\boldsymbol{\phi}_{\text{pairs}}$ and $\boldsymbol{\phi}_{\text{poly2}}$, corresponding to the kernel $K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\top \mathbf{x}_j)^2$. SBP hyperparameters: learning rate $= 10^{-5}$, iterations $= 20M$.

rule [13] or with the one-shot learning rule of the MHN, which is obtained by setting $\alpha^\mu = 1, \forall \mu$, in Eq. 4, that is

$$\mathbf{w} = \sum_{\mu}^{M} \xi_{\text{out}}^\mu \boldsymbol{\phi}(\boldsymbol{\xi}_{\text{in}}^\mu) \ . \tag{49}$$

Finally, we quantify the noise tolerance as the smallest Euclidean distance between the neuron's decision boundary and the patterns $\{\boldsymbol{\xi}_{\text{in}}^\mu\}_{\mu=1}^M$. This is equivalent to the minimum classification margin, defined as

$$\gamma = \min_{\mu} \frac{\xi_{\text{out}}^\mu (\mathbf{w}^\top \boldsymbol{\xi}_{\text{in}}^\mu)}{\|\mathbf{w}\|_2} \ . \tag{50}$$

We are only interested in the performance regime where all patterns are correctly recalled (i.e., correctly classified). This means that we only compare positive margins, since a negative margin indicates that there is one or more patterns that no longer can be correctly recalled. The results are plotted in Fig. C.1, and demonstrate that the margin for the MHN quickly drops with increasing load, while the iterative learning rule achieves a margin close to the theoretical optimum derived by Gardner [21]. Moreover, as the maximum storage capacity $M_{\max}$ of each learning rule can be found at the intersection between the margin curve and the line $\gamma = 0$, the capacity of the online rule can be estimated to $\sim 0.7N^2$, which is more than an order of magnitude higher than that of the MHN, which is $\sim 0.05N^2$.

D Generalized pseudoinverse rule

When the network is linear and underdetermined, meaning $M < N$, we can make sure that all patterns are attractors by modeling each neuron as a *least-squares SVM* [58] instead of a conventional SVM, so that the weights satisfy

$$\min_{\mathbf{w}} \|\mathbf{w}_i\|_2 \quad \text{s.t.} \quad \mathbf{w}_i^\top \boldsymbol{\xi}^\mu = \xi_i^\mu, \quad \forall \mu, i \ . \tag{51}$$

This is a minimum-norm interpolation problem, and yields the solution

$$\mathbf{s}^{(t+1)} = \text{sgn}\Big[\mathbf{X}\mathbf{K}^\dagger \mathbf{X}^\top \mathbf{s}^{(t)}\Big] = \text{sgn}\Big[\mathbf{X}\mathbf{X}^\dagger \mathbf{s}^{(t)}\Big] \tag{52}$$

where $\mathbf{K} = \mathbf{X}^\top \mathbf{X}$ is the *kernel matrix* and $\mathbf{K}^\dagger$ its Moore-Penrose pseudoinverse, and where we have used the property $\mathbf{K}^\dagger \mathbf{X}^\top = (\mathbf{X}^\top \mathbf{X})^\dagger \mathbf{X}^\top = \mathbf{X}^\dagger$. This is the *pseudoinverse learning rule* [48].

The derivation can be extended to MHNs by performing interpolation on the feature map $\boldsymbol{\phi}(\boldsymbol{\xi}^\mu)$. Assuming that the problem is still underdetermined, so that $M < N_\phi$, we aim to find the weights

$$\min_{\mathbf{w}} \|\mathbf{w}_i\|_2 \quad \text{s.t.} \quad \mathbf{w}_i^\top \boldsymbol{\phi}(\boldsymbol{\xi}^\mu) = \xi_i^\mu, \quad \forall \mu, i \tag{53}$$

which, analogously to the linear case, produces the optimal state update

$$\mathbf{s}^{(t+1)} = \operatorname{sgn}\Big[\mathbf{X}\mathbf{K}^\dagger K(\mathbf{X}, \mathbf{s}^{(t)})\Big] \tag{54}$$

where $\mathbf{K} = K(\mathbf{X}, \mathbf{X}) = \boldsymbol{\phi}(\mathbf{X})^\top \boldsymbol{\phi}(\mathbf{X})$. This can, again, be simplified to

$$\mathbf{s}^{(t+1)} = \operatorname{sgn}\Big[\mathbf{X}\boldsymbol{\phi}(\mathbf{X})^\dagger \boldsymbol{\phi}(\mathbf{s}^{(t)})\Big] \tag{55}$$

where we can identify the weight matrix $\mathbf{W} = \mathbf{X}\boldsymbol{\phi}(\mathbf{X})^\dagger$. This is the *generalized pseudoinverse learning rule*. Note that, if the feature-expanded patterns $\{\boldsymbol{\phi}(\boldsymbol{\xi}^\mu)\}_{\mu=1}^M$ are linearly independent, the kernel matrix is invertible and we have $\mathbf{K}^\dagger = \mathbf{K}^{-1}$.

E The kernel memory network for continuous patterns

E.1 Minimum norm interpolation and attractor basin size

Proof of Property 4. In the most general variant of this setting, each neuron i is modeled as a linear regressor with a neuron-specific feature map $\boldsymbol{\phi}_i$ and a state s_i which is updated according to

$$s_i^{(t+1)} = \mathbf{w}_i^\top \boldsymbol{\phi}_i(\mathbf{s}^{(t)}) \ . \tag{56}$$

All patterns $\mathbf{X}$ are guaranteed to be fixed points of the dynamics by finding the weights that satisfy

$$\xi_i^\mu = \mathbf{w}_i^\top \boldsymbol{\phi}_i(\boldsymbol{\xi}^\mu), \ \forall \mu, i \ . \tag{57}$$

In order for each pattern to also be an attractor, the weights must satisfy the additional constraint

$$\|\mathbf{J}_s\|_2 \Big|_{\mathbf{s}^{(t)} = \boldsymbol{\xi}^\mu} < 1, \ \forall \mu \tag{58}$$

where $\mathbf{J}_s$ is the Jacobian of the state update rule with respect to the input $\mathbf{s}^{(t)}$. The meaning of Eq. 58 is that the spectral norm of the Jacobian must be less than 1 when evaluated at each pattern. The reason for this is that the update rule, which computes $\mathbf{s}^{(t+1)}$ as a function of $\mathbf{s}^{(t)}$ (either synchronously or asynchronously) is continuously differentiable with respect to $\mathbf{s}^{(t)}$ and therefore satisfies the mean value inequality, so that

$$\left\|\mathbf{s}_1^{(t+1)} - \mathbf{s}_2^{(t+1)}\right\|_2 \leq \hat{J}_s \left\|\mathbf{s}_1^{(t)} - \mathbf{s}_2^{(t)}\right\|_2 \tag{59}$$

where $\hat{J}_s$ is an upper bound of the spectral norm, meaning

$$\|\mathbf{J}_s\|_2 \leq \hat{J}_s \ . \tag{60}$$

If $\hat{J}_s < 1$ at a pattern $\boldsymbol{\xi}^\mu$, it is also possible to find a neighborhood around $\boldsymbol{\xi}^\mu$ where $\hat{J}_s < 1$ holds as well, due to the continuity of the state update rule. Given this, the Banach fixed point theorem ensures that the state update rule is a contractive map in a region surrounding $\boldsymbol{\xi}^\mu$ and, equivalently, that $\boldsymbol{\xi}^\mu$ is a stable attractor [54]. While the complete basin of attraction of $\boldsymbol{\xi}^\mu$ might be difficult to compute exactly, we can define a subset of the basin as the set of points $\mathcal{S}^\mu$ in the open neighborhood of $\boldsymbol{\xi}^\mu$ satisfying

$$\mathcal{S}^\mu = \{\mathbf{s}^{(t)} : \|\mathbf{J}_s\|_2 < 1\} \ . \tag{61}$$

Given that the spectral norm of the Jacobian is upper bounded by the Frobenius norm, that is

$$\|\mathbf{J}_s\|_2 \leq \|\mathbf{J}_s\|_F \ , \tag{62}$$

we can obtain a lower bound of the extent of the basin of attraction with the set

$$\widehat{\mathcal{S}}^\mu = \{\mathbf{s}^{(t)} : \|\mathbf{J}_s\|_F < 1\} \ . \tag{63}$$

We can write $\mathbf{J}_s$ as

$$\mathbf{J}_s = \overline{\mathbf{W}} \cdot \overline{\mathbf{J}}_\phi \tag{64}$$

where

$$\overline{\mathbf{W}} = \begin{pmatrix} \mathbf{w}_1^\top & 0 & \cdots & 0 \\ 0 & \mathbf{w}_2^\top & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{w}_N^\top \end{pmatrix}, \quad \overline{\mathbf{J}}_\phi = \begin{pmatrix} \mathbf{J}_{\phi_1} \\ \mathbf{J}_{\phi_2} \\ \vdots \\ \mathbf{J}_{\phi_N} \end{pmatrix} \tag{65}$$

and where $\mathbf{J}_{\phi_i}$ is the Jacobian of $\boldsymbol{\phi}_i(\mathbf{s}^{(t)})$ with respect to $\mathbf{s}^{(t)}$. This gives us

$$\|\mathbf{J}_s\|_F = \|\overline{\mathbf{W}} \cdot \overline{\mathbf{J}}_\phi\|_F \leq \|\overline{\mathbf{W}}\|_F \cdot \|\overline{\mathbf{J}}_\phi\|_F \tag{66}$$

where the last expression is given by the Cauchy-Schwartz inequality. Since $\|\overline{\mathbf{J}}_\phi\|_F$ depends only on the kernel, which is fixed, the right-hand side of Eq. 66 can only be minimized by finding a set of weights that minimize $\|\overline{\mathbf{W}}\|_F$. By first rewriting this norm as

$$\|\overline{\mathbf{W}}\|_F^2 = \sum_i^N \|\mathbf{w}_i\|_2^2 \tag{67}$$

we see that its minimum is obtained by minimizing $\|\mathbf{w}_i\|_2, \forall i$. Combining this requirement with Eq. 57 is equivalent to performing a minimum norm interpolation, that is

$$\min_{\mathbf{w}_i} \|\mathbf{w}_i\|_2 \quad \text{s. t.} \quad \xi_i^\mu = \mathbf{w}_i^\top \boldsymbol{\phi}_i(\boldsymbol{\xi}^\mu), \quad \forall \mu, i \; . \tag{68}$$

If we now assume, as in the binary case, that all neurons use the same feature map, so that $\boldsymbol{\phi}_i = \boldsymbol{\phi}$, $\forall i$, the solution can be compactly written as in Eq. 20. This maximizes a lower bound of the attractor basin size, as defined by $\mathcal{S}^\mu$, for each pattern $\boldsymbol{\xi}^\mu$. □

E.2 Normally distributed patterns

E.2.1 Kernel at zero temperature

Proof of Property 5.1. Using the notation $\Delta = \|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2$, we have that

$$\lim_{\beta\to\infty} \left(\frac{\Delta}{r}\right)^\beta = \begin{cases} 0 \; , & \Delta < r \\ 1 \; , & \Delta = r \\ \infty \; , & \Delta > r \end{cases} \tag{69}$$

from which it follows that

$$\lim_{\beta\to\infty} \exp\left[-\left(\frac{\Delta}{r}\right)^\beta\right] = \begin{cases} 1 \; , & \Delta < r \\ e^{-1} \; , & \Delta = r \\ 0 \; , & \Delta > r \end{cases} \tag{70}$$

which is equivalent to $\Theta(r - \Delta)$ with $\Theta(0) = e^{-1}$. We combine this with the assumption that the patterns are unique and that $\min_{\mu,\nu\neq\mu} \|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2 > r$ and obtain

$$\lim_{\beta\to\infty} K_{\exp_\beta}(\boldsymbol{\xi}^\mu, \boldsymbol{\xi}^\nu) = \Theta(r - \|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2) = \begin{cases} 1 \; , & \mu = \nu \\ 0 \; , & \mu \neq \nu \end{cases} \tag{71}$$

which can be written compactly as

$$\lim_{\beta\to\infty} \mathbf{K}_{\exp_\beta} = \mathbf{I}_M \; . \tag{72}$$

It directly follows that

$$\lim_{\beta\to\infty} \mathbf{K}_{\exp_\beta}^{-1} = \mathbf{I}_M^{-1} = \mathbf{I}_M \tag{73}$$

and, therefore,

$$\lim_{\beta\to\infty} \mathbf{X}\mathbf{K}_{\exp_\beta}^{-1} K_{\exp_\beta}(\mathbf{X}, \mathbf{s}^{(t)}) = \mathbf{X}\,\Theta(r^2 - \|\mathbf{X} - \mathbf{s}^{(t)}\|_2^2) \; . \tag{74}$$

□

E.2.2 Noise robustness

Proof of Property 6. With $\mathbf{s}^{(0)} = \boldsymbol{\xi}^\mu + \boldsymbol{\epsilon}$, we have

$$\|\boldsymbol{\xi}^\mu - \mathbf{s}^{(0)}\|_2^2 = \|\boldsymbol{\epsilon}\|_2^2 = \|\sigma\boldsymbol{\epsilon}_0\|_2^2 = \sigma^2\|\boldsymbol{\epsilon}_0\|_2^2 \tag{75}$$

where $\boldsymbol{\epsilon}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$, from which it follows that $\|\boldsymbol{\epsilon}_0\|_2^2$ is a random variable with a $\chi^2(N)$ distribution. According to the central limit theorem, we also have

$$\lim_{N\to\infty} \frac{\|\boldsymbol{\epsilon}_0\|_2^2 - N}{\sqrt{2N}} \sim \mathcal{N}(0,1) \tag{76}$$

where we have used the fact that each term in $\|\boldsymbol{\epsilon}_0\|_2^2$ is $\chi^2(1)$-distributed, and has mean 1 and variance 2. We will therefore make the approximation $\|\boldsymbol{\epsilon}_0\|_2^2 \sim \mathcal{N}(N, 2N)$ for large N. This gives us

$$\sigma^2\|\boldsymbol{\epsilon}_0\|_2^2 \sim \mathcal{N}(\sigma^2 N, 2\sigma^4 N)\ . \tag{77}$$

The original pattern $\boldsymbol{\xi}^\mu$ will only be recovered if $r^2 - \|\boldsymbol{\xi}^\mu - \mathbf{s}^{(0)}\|_2^2 \geq 0$, which is satisfied in at least 50% of trials if $r^2 \geq \sigma^2 N$. The maximum variance with which this type of recovery still holds is thus

$$\sigma^2_{\max} = r^2/N\ . \tag{78}$$

□

E.2.3 Storage capacity

Proof of Property 7. In the limit $\beta \to \infty$, the boundary of the basin of attraction surrounding each pattern is sharp. In this setting, we are guaranteed that each pattern can be recalled without errors as long as $\min_{i,j\neq i}\|\boldsymbol{\xi}^i - \boldsymbol{\xi}^j\|_2 > 2r$. We will therefore estimate the storage capacity by calculating the number of patterns, on average, that can be loaded into the network before at least two attractor basins overlap and the condition above is violated (see Fig. E.1).

We begin by observing that for two random patterns $\boldsymbol{\xi}^i, \boldsymbol{\xi}^j \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$, we have

$$\frac{1}{2}\|\boldsymbol{\xi}^i - \boldsymbol{\xi}^j\|_2^2 \sim \chi^2(N) \tag{79}$$

which, using the central limit theorem as in Eq. 76, can be approximated as $\mathcal{N}(N, 2N)$ for large N, thereby yielding

$$\|\boldsymbol{\xi}^i - \boldsymbol{\xi}^j\|_2^2 \sim \mathcal{N}(2N, 8N)\ . \tag{80}$$

We now assume that the squared Euclidean distance between each pair of patterns $\boldsymbol{\xi}^i, \boldsymbol{\xi}^j$ in a set of M given patterns $\{\boldsymbol{\xi}^\mu\}_{\mu=1}^M$ is an independent sample of a random variable, denoted Δ^2, which is distributed as in Eq. 80. This is, of course, an approximation which neglects that the pairwise distances between any set of points are inter-dependent. Nonetheless, for relatively large N and M, the approximation accurately describes the empirical distance distribution.

Relying on this assumption, the process of drawing M random patterns becomes equivalent to drawing $M(M-1)/2$ unique pairwise distances Δ^2 from the distance distribution. The probability of drawing a sample $\Delta^2 \leq 4r^2$ can be calculated using the cumulative distribution function for the standard normal distribution, given by $\Phi(x) = \frac{1}{2}\operatorname{erfc}(-x)$, according to

$$\mathbb{P}(\Delta^2 \leq 4r^2) = \frac{1}{2}\operatorname{erfc}\left(\frac{N - 2r^2}{2\sqrt{N}}\right)\ . \tag{81}$$

The average number of samples of Δ^2 one needs to draw before a sample satisfies $\Delta^2 \leq 4r^2$ is given by $\mathbb{P}(\Delta^2 \leq 4r^2)^{-1}$. This determines the maximum number of patterns that the network, on average, can store, according to

$$\frac{M_{\max}(M_{\max}-1)}{2} = \frac{1}{\mathbb{P}(\Delta^2 \leq 4r^2)}\ . \tag{82}$$

We combine this expression with the approximation $M_{\max}(M_{\max}-1) \approx M_{\max}^2$ (which holds for large $M_{\max}$) and Eq. 81, and obtain

$$M_{\max} = 2\operatorname{erfc}\left(\frac{N - 2r^2}{2\sqrt{N}}\right)^{-1/2}\ . \tag{83}$$

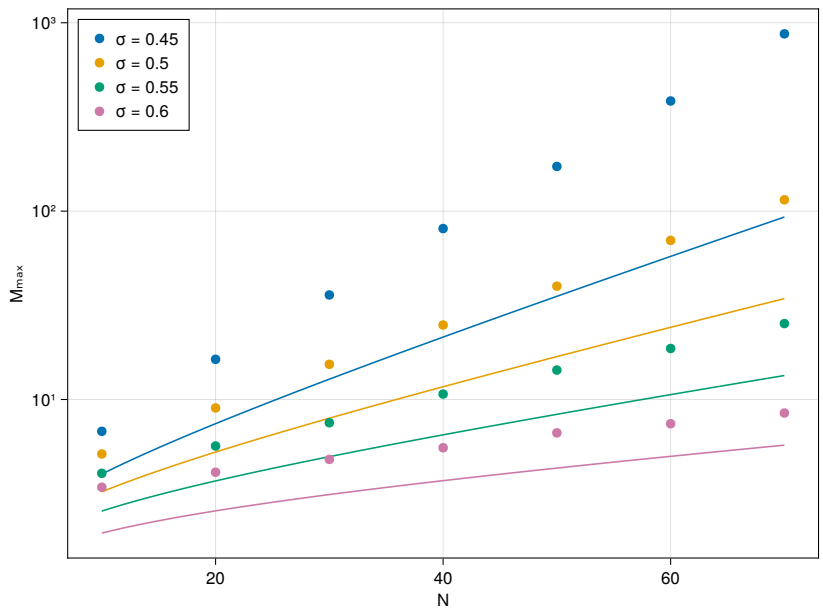

Figure E.1: Plot of the storage capacity of the Exp_β network at $\beta \to \infty$ with normally distributed patterns. Dots represent means ($\pm$ s.e.m.) over 1000 simulations, in which the capacity is determined by the number of patterns sampled until one of the pairwise distances is smaller than $2r$. The standard error is too small to be visible. Lines correspond to the bound in Eq. 87. Note that the plot is log-linear, so the linear increase indicates that $M_{\max}$ scales exponentially in N.

We now parameterize the radius r in terms of the largest tolerable noise amplitude, according to Eq. 78. This gives us

$$M_{\max} = 2\,\text{erfc}\left(\frac{\sqrt{N}(1-2\sigma_{\max}^2)}{2}\right)^{-1/2} . \tag{84}$$

Given that the erfc function can be well approximated using the asymptotic expansion

$$\text{erfc}(x) \approx \frac{e^{-x^2}}{\sqrt{\pi}x}\sum_{n=0}^{\infty}(-1)^n\frac{(2n-1)!!}{(2x^2)^n} \tag{85}$$

for large arguments x, we can obtain a tight lower bound of erfc^{-1} as long as N is large and $\sigma_{\max}^2 \lesssim 1/2$ with the inverse zeroth order expansion, thereby obtaining

$$\text{erfc}(x)^{-1} \geq \sqrt{\pi}xe^{x^2} . \tag{86}$$

We insert Eq. 86 in 84 and finally obtain

$$M_{\max} \geq \sqrt{2\sqrt{\pi N}(1-2\sigma_{\max}^2)}\exp\left[\frac{N(1-2\sigma_{\max}^2)^2}{8}\right] . \tag{87}$$

□

E.3 Patterns on the hypersphere

E.3.1 Storage capacity

Property 8 (Storage capacity: patterns on the hypersphere)**.** *At $\beta \to \infty$, the average maximum number of patterns that the Exp_β network can store and recall without errors is lower-bounded by*

$$M_{\max} \geq \sqrt{\sqrt{8\pi N}(1-2r^2)}\exp\left[\frac{N(1-2r^2)^2}{4}\right] \tag{88}$$

when each pattern is randomly drawn from $\mathbb{S}^{N-1}$.

Proof. We begin by observing that for two random patterns $\boldsymbol{\xi}^i, \boldsymbol{\xi}^j \in \mathbb{S}^{N-1}$, we have

$$\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2^2 = 2(1 - \boldsymbol{\xi}^{\mu\top}\boldsymbol{\xi}^\nu) \ . \tag{89}$$

The probability distribution for the inner product $\omega = \boldsymbol{\xi}^{\mu\top}\boldsymbol{\xi}^\nu$ can be found in [60], and is given by

$$\omega \sim \frac{1}{\sqrt{\pi}} \frac{\Gamma(\frac{N}{2})}{\Gamma(\frac{N-1}{2})} (1-\omega^2)^{\frac{N-3}{2}} \tag{90}$$

which, for large N, can be approximated as

$$\omega \sim \mathcal{N}(0, \frac{1}{N}) \ . \tag{91}$$

We use Eq. 91 in 89 and obtain

$$\Delta^2 \sim \mathcal{N}(2, \frac{4}{N}) \ . \tag{92}$$

The probability of placing a pair of random points on $\mathbb{S}^{N-1}$ with $\Delta^2 \leq 4r^2$ is thus

$$\mathbb{P}(\Delta^2 \leq 4r^2) = \frac{1}{2} \operatorname{erfc}\left(\frac{\sqrt{N}(1-2r^2)}{\sqrt{2}} \right) \ . \tag{93}$$

The average number of samples of Δ^2 one needs to draw before a sample satisfies $\Delta^2 \leq 4r^2$ is given by $\mathbb{P}(\Delta^2 \leq 4r^2)^{-1}$, and the maximum number of patterns that the network, on average, can store, is therefore

$$M_{\max} = 2 \operatorname{erfc}\left(\frac{\sqrt{N}(1-2r^2)}{\sqrt{2}} \right)^{-1/2} \ . \tag{94}$$

We use the lower bound in Eq. 86 in 94 and finally obtain

$$M_{\max} \geq \sqrt{\sqrt{8\pi N}(1-2r^2)} \exp\left[\frac{N(1-2r^2)^2}{4} \right] \ . \tag{95}$$

□

E.4 Bipolar patterns

E.4.1 Noise robustness

Property 9.1 (Robustness to flipped bits)**.** *Assume that we are given a set of unique patterns $\boldsymbol{\xi}^1, \dots, \boldsymbol{\xi}^M \in \{\pm 1\}^N$ with $\min_{\mu,\nu\neq\mu} \|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2 > 2r$, and that the Exp$_\beta$ network is initialized in a distorted pattern $\mathbf{s}^{(0)} = \boldsymbol{\xi}^\mu \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \in \{\pm 1\}^N$, with $\mathbb{P}(\epsilon_i = -1) = \rho, \forall i$. Then, at $\beta \to \infty$, the maximum bit-wise error probability $\rho_{\max}$ with which $\boldsymbol{\xi}^\mu$ can be recovered in at least 50% of trials is*

$$\rho_{\max} = r^2/4N \ . \tag{96}$$

Proof. With $\mathbf{s}^{(0)} = \boldsymbol{\xi}^\mu \odot \boldsymbol{\epsilon}$, we have

$$\|\boldsymbol{\xi}^\mu - \mathbf{s}^{(0)}\|_2^2 = \|2\boldsymbol{\epsilon}_{\mathcal{B}}\|_2^2 = 4\|\boldsymbol{\epsilon}_{\mathcal{B}}\|_2^2 \tag{97}$$

where $\boldsymbol{\epsilon}_{\mathcal{B}} \in \{0,1\}^N$, with each entry being a random variable distributed as $(\boldsymbol{\epsilon}_{\mathcal{B}})_i \sim \text{Bernoulli}(\rho)$. This implies that $\|\boldsymbol{\epsilon}_{\mathcal{B}}\|_2^2 \sim \text{Binomial}(N, \rho)$, which can be approximated as $\mathcal{N}(\rho N, \rho(1-\rho)N)$ for large N. This gives

$$4\|\boldsymbol{\epsilon}_{\mathcal{B}}\|_2^2 \sim \mathcal{N}(4\rho N, 16\rho(1-\rho)N) \ . \tag{98}$$

Again, the original pattern $\boldsymbol{\xi}^\mu$ will only be recovered if $r^2 - \|\boldsymbol{\xi}^\mu - \mathbf{s}^{(0)}\|_2^2 \geq 0$, which is satisfied in at least 50% of trials if $r^2 \geq 4\rho N$. The maximum bit-wise error probability with which this type of recovery still holds is thus

$$\rho_{\max} = r^2/4N \ . \tag{99}$$

□

In Eq. 78, σ roughly quantifies the maximum noise fluctuation around a pattern that is tolerable with a given radius r, while still being able to recover the pattern in a majority of trials. In the case of Eq. 99, ρ instead quantifies the maximum tolerable bit-wise error probability.

E.4.2 Storage capacity

Property 9.2 (Storage capacity: bipolar patterns)**.** *At $\beta \to \infty$, the average maximum number of bipolar patterns with sparseness f that the Exp$_\beta$ network can store and recall without errors is lower-bounded by*

$$M_{\max} \geq 2\left(\frac{\pi N}{2\tilde{f}(1-\tilde{f})}\right)^{1/4}\left(\tilde{f}-4\rho_{\max}\right)^{1/2}\exp\left[\frac{N(\tilde{f}-4\rho_{\max})^2}{4\tilde{f}(1-\tilde{f})}\right] \tag{100}$$

where $\tilde{f} = 2f(1-f)$ and $\rho_{\max}$ is the maximum bit-wise error probability tolerated by the network.

Proof. This proof is, again, a slightly modified variant of the proof of Property 7. First, we observe that for two random patterns $\boldsymbol{\xi}^\mu, \boldsymbol{\xi}^\nu \in \{\pm 1\}^N$ with sparseness f, so that $\mathbb{P}(x_i^{\mu,\nu}=1) = f$, it is true that

$$\frac{1}{4}\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2^2 \sim \text{Binomial}(N, \tilde{f}) \tag{101}$$

where $\tilde{f} = 2f(1-f)$ denotes the probability that $\boldsymbol{\xi}^\mu$ and $\boldsymbol{\xi}^\nu$ differ at any given entry. For large N, we can again approximate $\text{Binomial}(N, \tilde{f})$ with $\mathcal{N}(\tilde{f}N, \tilde{f}(1-\tilde{f})N)$, which gives us

$$\|\boldsymbol{\xi}^\mu - \boldsymbol{\xi}^\nu\|_2^2 \sim \mathcal{N}(4\tilde{f}N, 16\tilde{f}(1-\tilde{f})N)\ . \tag{102}$$

We use this to compute the upper bound of the probability of drawing a distance Δ^2 which satisfies $\Delta^2 \leq 4r^2$, as in Eq. 81. The result is

$$\mathbb{P}(\Delta^2 \leq 4r^2) = \frac{1}{2}\,\text{erfc}\left(\frac{\tilde{f}N - r^2}{\sqrt{2\tilde{f}(1-\tilde{f})N}}\right)\ . \tag{103}$$

Following the same derivations used to produce Eqs. 82 and 83, we arrive at

$$M_{\max} = 2\,\text{erfc}\left(\frac{\tilde{f}N - r^2}{\sqrt{2\tilde{f}(1-\tilde{f})N}}\right)^{-1/2}\ . \tag{104}$$

As before, we parameterize the radius r in terms of the maximum tolerable bit-wise error probability according to Eq. 99 and yield

$$M_{\max} = 2\,\text{erfc}\left(\frac{\sqrt{N}(\tilde{f}-4\rho_{\max})}{\sqrt{2\tilde{f}(1-\tilde{f})}}\right)^{-1/2}\ . \tag{105}$$

We finally replace erfc with the lower bound in Eq. 86. This is valid for large N and when $\rho \lesssim \tilde{f}/4$. We obtain

$$M_{\max} \geq 2\left(\frac{\pi N}{2\tilde{f}(1-\tilde{f})}\right)^{1/4}\left(\tilde{f}-4\rho_{\max}\right)^{1/2}\exp\left[\frac{N(\tilde{f}-4\rho_{\max})^2}{4\tilde{f}(1-\tilde{f})}\right]\ . \tag{106}$$

□

F Comparison to neuron models with active dendrites

To demonstrate how single neurons in kernel memory networks are generalizations of abstract neuron models with active dendrites, we begin by considering a neuron in the feature form (Eq. 5). We will here use the Heaviside activation function with threshold θ, denoted Θ_θ, instead of sgn, and assume that patterns and states are in $\{0,1\}^N$, where N is the number of inputs. This, however, does not change the fundamental properties of our model, as SVMs can be formulated for binary data with minor modifications. Assuming a polynomial feature map $\boldsymbol{\phi}$ of degree p, the feature vector will

consist of all possible monomials of degree $\leq p$ composed of the states of its input neurons. Setting, for example, $p = 2$ gives us

$$\begin{aligned}\boldsymbol{\phi}(\mathbf{x}) = (&1, \sqrt{2}x_1, \ldots, \sqrt{2}x_N, \\ &\sqrt{2}x_1x_2, \ldots, \sqrt{2}x_1x_N, \sqrt{2}x_2x_3, \ldots, \sqrt{2}x_{N-1}x_N, \\ &x_1^2, \ldots, x_N^2)\ .\end{aligned} \tag{107}$$

By limiting the feature map to only include a subset of all terms, our model is reduced to the 2-degree *sigma-pi unit* [56, p. 73], which can be written as

$$s_{\text{out}} = \Theta_\theta \left[\sum_i w_i \prod_{j \in \mathcal{C}_i} s_{\text{in},j} \right] \tag{108}$$

where w_i is the weight of cluster i, which consists of a product of all inputs $s_{\text{in},j}$ whose indices j are contained in the set $\mathcal{C}_i$. From a neurophysiological perspective, each product represents the cross-talk between a set of synapses. By including such multiplicative interactions, synapses can both gate and amplify each other, to generate supra-linear input currents.

If we now further constrain this model to include only a subset of the cross-terms x_ix_j, $i \neq j$, and parameterize each cross-term weight as $w_{ij} = w_iw_j$, our neuron model is reduced to the *clusteron* [42], which can be written as

$$\begin{aligned} s_{\text{out}} &= \Theta_\theta \left[\sum_i^N \sum_{j \in \mathcal{C}_i} w_iw_js_{\text{in},i}s_{\text{in},j} \right] \\ &= \Theta_\theta \left[\sum_i^N w_is_{\text{in},i} \left(\sum_{j \in \mathcal{C}_i} w_js_{\text{in},j} \right) \right] \end{aligned} \tag{109}$$

where $\mathcal{C}_i$ now is the set describing all inputs j that input i should be paired with.

We now consider a neuron in the kernel form (Eq. 6) with an inner-product kernel $K(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i^\top \mathbf{x}_j)$. By setting $\xi_{\text{out}}^\mu = 1, \forall \mu$, and assuming binary inputs, our model is reduced to

$$s_{\text{out}} = \Theta_\theta \left[\sum_\mu^M \alpha^\mu k \left(\sum_i^N \xi_{\text{in},i}^\mu s_{\text{in},i} \right) \right] \tag{110}$$

which is equivalent to the pyramidal cell as a 2-layered neural network, as defined in [51]. According to the original interpretation, this neuron model is comprised of M subunits, which can represent, for example, separate parts of a dendritic tree. All subunits receive the inputs and produce separate outputs which are all summed in the soma. Each subunit μ is characterized by the input weights $\boldsymbol{\xi}_{\text{in}}^\mu$ (which serves as a mask), the output weight α^μ (which determines how strongly the subunit influences the response at the soma), and the activation function k.