One-shot learning and robust recall with BTSP, a biological synaptic plasticity rule

Yujie Wu and Wolfgang Maass†

Institute of Theoretical Computer Science, Graz University of Technology, Graz, Austria
†Corresponding author. E-mail: yujie.wu@igi.tugraz.at; maass@igi.tugraz.at

ABSTRACT

Recent experimental studies in the awake brain have identified a rule for synaptic plasticity that is instrumental for the instantaneous creation of memory traces in area CA1 of the mammalian brain: Behavioral Time scale Synaptic Plasticity (BTSP). This one-shot learning rule differs in essential aspects from previously considered plasticity mechanisms. We introduce a transparent model for the core function of BTSP, and establish a theory that enables a principled understanding of the system of memory traces that it creates. Our theoretical predictions and numerical simulations show that BTSP is able to create a functionally powerful content-addressable memory without the need for high-resolution synaptic weights. Furthermore, it reproduces the repulsion effect of human memory, whereby traces for similar memory items are pulled apart to enable differential downstream processing. Altogether, our results create a link between synaptic plasticity in area CA1 of the hippocampus and its network function. They also provide a blueprint for implementing content-addressable memory with on-chip learning capability in highly energy-efficient crossbar arrays of memristors.

Introduction

The brain has to solve a really challenging algorithmic problem when it stores episodic memories: It has to allocate on-the-fly neurons that form a trace or tag for a new experience that can subsequently be reactivated with partial cues. In other words, it needs a mechanism for creating a content-addressable memory (CAM) through one-shot learning. Recent experimental work has elucidated an essential ingredient of that: Behavioral time scale synaptic plasticity (BTSP) (Bittner et al., 2017; Magee and Grienberger, 2020; Milstein et al., 2021). BTSP differs strongly from plasticity rules such as the Hebb rule and STDP (spike-timing-dependent plasticity) that have previously been considered in efforts to model the creation of memory traces in the brain. This is especially salient because in contrast to most experimental data on synaptic plasticity (Chéreau et al., 2022), BTSP has been validated through experiments in awake and behaving animals. Other converging experimental data suggest that area CA1 of the hippocampus, where BTSP has first been demonstrated, is a brain area that is central for the creation of memory traces for episodic and conjunctive memories (Sugar and Moser, 2019).

BTSP differs in the following aspects from traditionally studied plasticity rules:

- BTSP does not depend on the firing of the postsynaptic neuron.
- Instead, it is gated by synaptic input from another brain area, the entorhinal cortex (EC). These gating signals appear to be largely stochastic (Grienberger and Magee, 2022).
- BTSP does not require dozens of repetitions of a protocol for the induction of synaptic plasticity, but is effective in a single or few trials, i.e., it provides a mechanism for one-shot learning (Bittner et al., 2017).

- The direction in which BTSP changes a synaptic weight depends primarily on the preceding weight value (Milstein et al., 2021).

- BTSP acts on the time scale of seconds, a time scale that is suitable for creating episodic and other forms of conjunctive memories that integrate temporally dispersed information.

A first mathematical model for BTSP and its impact on the induction of place cells was provided by (Milstein et al., 2021). There, and in the subsequent modelling studies of (Cone and Shouval, 2021; Li and Roxin, 2023) for the induction of place cells, BTSP was modelled through a differential equation with continuous time and weights, but without noise. The latter is noteworthy since the raw experimental data for BTSP, see Fig. 3C of (Milstein et al., 2021), indicate a substantial amount of trial-to-trial variability of weight changes. Furthermore, it is unlikely that the size of the dendritic spine on which a synapse is located, the standard proxy for the strength of its synaptic weight (Bartol Jr et al., 2015), can be finely adjusted during a single or even a few quickly repeating trials. Hence, it is plausible that measurements of seemingly continuous-valued changes of synaptic weights during BTSP result from a superposition of coarser and partially stochastic synaptic changes at multiple synaptic connections. This gives rise to the question whether there exists a simple rule for such coarser and partially stochastic synaptic changes at single synaptic release sites that could model experimental data on BTSP. The first simple rule for BTSP with binary weights had been proposed in a preliminary version of the present study (Wu and Maass, 2023). Subsequently, a similar rule was also proposed as a model for new experimental data on BTSP for recurrent connections within area CA3 (Li et al., 2023).

We will examine here both the rule from (Wu and Maass, 2023) and a stochastic variant of it, and create a theoretical framework for studying the impact of these rules on the network level. We show that more complex BTSP rules with continuous weights can be approximated through the superposition of this simple BTSP rule with binary weights at multiple synaptic release sites between pre- and postsynaptic neurons. Furthermore, we investigate the functional properties of memory systems that are created by simple BTSP rules with binary weights, to which we refer as default in the following as BTSP rules. We find that BTSP is especially suited for creating content-addressable memory systems (CAM) for sparsely encoded memory items. This is insofar surprising as the most commonly studied CAM model, the Hopfield network (HFN), requires weights that can assume a large number of different values. If one reduces the resolution of their weights to just a few bits, their memory capacity drops drastically (Tarkov, 2019). Hence it is not clear whether HFNs provide the right framework for modelling CAM in the brain, since biological synapses can apparently assume only a relatively small number of distinct values (Bartol Jr et al., 2015). For one-shot learning of memories, the biological repertoire of weight values is likely to be even smaller. Similarly, the HFN does not provide a satisfactory blue-print for CAM in neuromorphic hardware if one wants to produce energy-efficient CAM in crossbar arrays of inexpensive memristors that can only assume a small number of distinguishable resistance states.

The brain employs a sparse coding regime, where relatively few neurons are simultaneously active. This sparse coding regime contributes to its astoundingly low energy consumption, and is therefore also desirable for neuromorphic hardware. Consequently, we focus our investigation on the creation of CAM for memory items that are represented through brain-like sparse activity. We show through theoretical analysis and numerical simulations that BTSP is able to create a CAM with just binary synaptic weights for sparse codes that achieves about the same performance level as a HFN with continuous-valued weights. In fact, it exhibits functional advantages over a HFN with continuous-valued weights since it allows stable recall also for memory items that are not independent or orthogonal. Orthogonality of patterns that are to be stored is an inherent requirement of HFNs, but this requirement is in conflict with the need to create distinguishable memory traces for memory items that have common content, such as two episodes that are similar except for a few salient features, or conjunctive memories that contain common components. In fact, the human brain can not only create different memory traces for similar memory items, but memory traces for similar memory items exhibit a repulsion effect (Chanales et al., 2017; Wanjia et al., 2021; Zheng et al., 2021). This effect,

which could so far not be reproduced by HFNs or other learning rules, pulls memory traces for similar memory items actively apart, thereby enabling differential downstream processing. We show that BTSP can reproduce this repulsion effect of memory traces in the human brain.

An especially interesting facet of experimental data on BTSP is its use of stochastic gating signals from area EC. This stochastic feature of BTSP can be seen as an extension of a well-known biological algorithm, the so-called fly algorithm (Dasgupta et al., 2017), and also as an extension of random projections, a well-known method for the creation of memory traces in digital computers. More precisely, one can view BTSP as a means to port advantages of stochastic neural coding from a small neural system, the olfactory system of the fly, into the domain of learning and memory on a larger neural system. The fly algorithm also employs binary weights, but these are assumed to exist a-priori, they are not learnt. In contrast, BTSP produces a random projection from input patterns to memory traces through learning. In other words, the random projection that BTSP creates is custom-made for a particular ensemble of input patterns, and can be continually extended through one-shot learning. We show that this difference to the fly algorithm, and to random projections, in general, enables BTSP to create attractors around stored memory traces that enable stable recall from partial cues. In addition, BTSP manages to overcome two bottlenecks of applying the fly algorithm to large systems: It does not require a projection of input patterns into a 40 times larger neural population, nor a global winner-take-all competition over the population of neurons into which input patterns are projected. These two requirements are met by the relatively small olfactory system of the fly, but not by the memory system of the mammalian brain. The number of neurons in area CA1, the area into which memory items are projected from area CA3, is at most by a factor of 1.5 larger than the area CA3, both in the rodent and in the human brain (see Table 3-1 in (Andersen et al., 2007)). Furthermore, area CA1 consists of 0.39 to 14 million neurons in the mammalian brain (Andersen et al., 2007), and the assumption of a winner-take-all computation is more problematic for such a large system.

Since BTSP is a local synaptic plasticity rule that requires in its simplified form only two weight values, it is especially suited for on-chip learning in neuromorphic hardware, in particular for online creation and expansion of CAM in crossbar arrays of memristors with just two required resistance states. Hence this new CAM paradigm provides a blue-print for highly energy-efficient large-scale implementations of CAMs for brain-like sparse patterns. This approach can build on and drastically expand the memory capacity of already existing implementations of CAM in crossbar arrays of memristors that were based on the HFN paradigm (Hu et al., 2015; Sun et al., 2021; Wang et al., 2020).

We review experimental data on BTSP in the next section and show that a simple version of the rule with binary weights provides a good approximation to these data. We then address the main new parameter of this rule, the probability of a stochastic gating signal, and elucidate its functional impact. In the subsequent section, we compare the properties of the system of memory traces that are created through BTSP with the memory systems that are created by random projections, the main component of the fly algorithm. Subsequently, we present theory and numerical simulation results for the recall of memory traces for non-orthogonal memory items for BTSP and HFNs. We then compare the properties of the CAMs that are created through BTSP with feedback connections with properties of CAMs that are implemented by HFNs, both for the case of binary and continuous-valued synaptic weights. Finally, we show that BTSP reproduces the repulsion effect of the human brain, and elucidate parameters of BTSP that are critical for that.

Results

A transparent rule captures core features of BTSP. We first recapitulate experimental data on BTSP, and then discuss options for modelling it by simple rules. Memory tags, also referred to as memory traces, are created in a few or even a single trial through the plasticity of synaptic connections from pyramidal cells in areas CA3 to

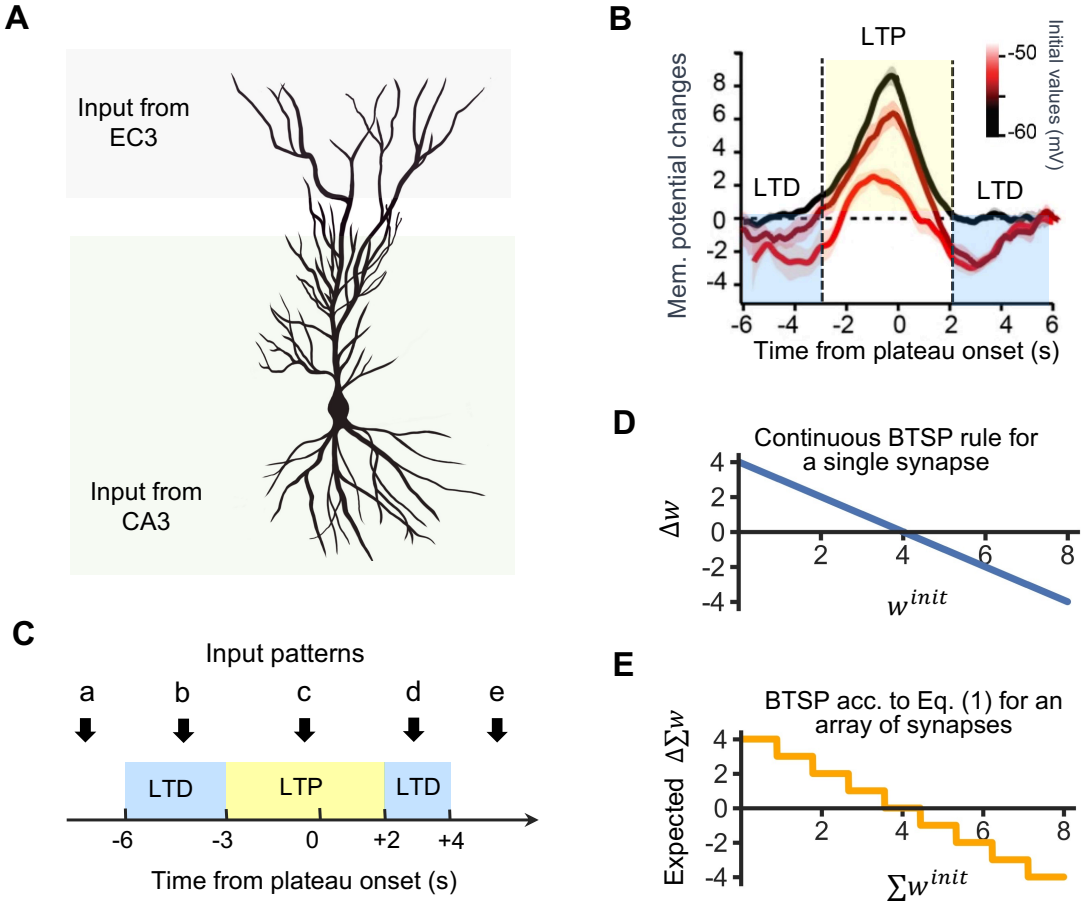

Figure 1. Experimental data and simplified rules for BTSP. (**A**) A CA1 pyramidal cell gets synaptic inputs from at least two different sources: One stream of synaptic inputs arrives in distal parts of apical dendrites from layer 3 of the entorhinal cortex (EC3) and induces plateau potentials (Grienberger and Magee, 2022). Another stream arrives from area CA3 at more proximal synapses, see Fig.5-1 in (Andersen et al., 2007). (**B**) According to the averages of experimental data shown in Fig.3E of (Milstein et al., 2021) the change of synaptic weights through BTSP depends on the initial weight and the time of presynaptic firing relative to the onset of a plateau potential (= time 0), Resulting weight changes are shown for 3 different ranges of the initial synaptic strength (black: minimal initial strength, orange: maximal initial strength, dark red: intermediate initial strength). Synaptic strength was measured in terms of the impact of synaptic input on the postsynaptic membrane potential. This panel is redrawn from Fig.3E of (Milstein et al., 2021). (**C**) Resulting scheme for the decision between LTP and LTD. According to the experimental data reported in Fig. 3 E of (Milstein et al., 2021) only input patterns that arrive up to 6s before or at most 4s after the initiation of a plateau potential in the CA1 neuron trigger synaptic plasticity through BTSP. Hence BTSP would apply in this drawing to patterns b, c and d, but not to patterns a or e. In addition, LTP or LTD are only applied when the synapse has the right initial weight. (**D**) Weight changes in dependence of the initial weight for a more complex biophysical model of BTSP according to Eq. 1 in (Milstein et al., 2021), where the initial weight can assume continuous values. We consider here the case where the input patterns arrive at the same time as the plateau potential. (**E**) Demonstration that the simpler version of the BTSP rule according to Eq.1 produces similar expected net changes of the strength of a connection between a pre- and postsynaptic neuron in case they are connected by 8 synaptic release sites, each of them with binary weights that are subject to the BTSP rule of Eq. 1.

pyramidal cells in area CA1 of the hippocampus. Fig. 1A provides a schematic drawing of a pyramidal cell in area CA1, which receives synaptic input from area CA3, as well as synaptic input from layer 3 of EC on distal dendrites, which create plateau potentials in its apical tuft. Experimental data from (Bittner et al., 2015, 2017; Grienberger and Magee, 2022) show that these plateau potentials gate the plasticity of weights of synaptic connections from CA3 to pyramidal cells in area CA1. Since these plateau potentials, and their impact on synaptic plasticity, last for several seconds, the resulting plasticity process is termed behavioral time scale synaptic plasticity.

More precisely, a plateau potential opens the plasticity window for several seconds before and after its onset (= time 0 in Fig. 1B). Experimental data on the resulting weight change exhibit substantial trial-to-trial variability, see Fig. 3B of (Milstein et al., 2021). But according to Fig. 1B, which is redrawn after Fig. 3E of (Milstein et al., 2021), one sees that the decision between LTP (long-term potentiation), LTD (long-term depression), and no weight change, depends on average on two factors: the prior value of the synaptic weight and the time difference of presynaptic firing and the onset of the plateau potential. Importantly, in contrast to Hebbian plasticity and STDP, BTSP does not depend on the firing of the CA1 neuron. The black curve in Fig. 1B shows that LTP occurs for synapses whose prior strength was small, provided that the synaptic input arrives within the interval from -3 to 2s relative to the onset of the plateau potential. Outside of this time interval the weights of these synapses remain largely unchanged. The orange curve shows that LTD is induced for strong synapses, provided that the synaptic input arrives within the windows from -6 to -3s or from 2 to 4s relative to the onset of the plateau potential. If the synaptic input arrives in between, the resulting weight change is negligible. Thus the dependence of the decision between LTP and LTD on the relative timing between the arrival of the synaptic input and the onset of the plateau potential can be summarized by the simple scheme of Fig. 1C. Hence, the chance that a synaptic input that arrives within the window of plasticity marked by a plateau potential falls into the LTP window is about 0.5, and the chance that it falls into the LTD window is also 0.5, provided there is no systematic relation between the firing times of presynaptic neurons in areas CA3 and layer 3 of EC. Apart from that, one should keep in mind that even if the synaptic input falls into the right time window, LTP is applied primarily to weak synapses and LTD primarily to strong synapses.

The dark red 3rd curve in Fig. 1B shows weight changes for the case of intermediate initial weight values, indicating a similar dependence of the decision between LTP and LTD on relative timing. In general, there are actually several synaptic release sites between synaptically connected neurons, and an intermediate amplitude of the postsynaptic potential could also result from a superposition of postsynaptic potentials from both strong and weak connections at different synaptic release sites. In fact, it is not known whether the single-trial plasticity under BTSP can modulate the strengths of individual synaptic release sites in a continuous manner. This reasoning justifies the arguably simplest model for BTSP with binary synaptic weights, as defined by the black and orange curves in Fig. 1B. We will focus on this simple model in the following because it not only captures the core features of experimental data on BTSP but also is analytically tractable. Thus, we approximate the experimental data on BTSP by the following simple rule: When an input pattern $\boldsymbol{x}$ arrives within the time interval from -6 to 4s relative to the onset of a plateau potential, synaptic weights of the neuron are increased by BTSP with probability 0.5 (provided they are not yet at their maximal value of 1), and also decreased with probability 0.5 (provided they are not yet at their minimal value of 0). Hence, with a probability 0.5 the following rule is applied to all synapses of the neuron:

$$\Delta w_i = \begin{cases} +1 & \text{if } x_i = 1 \text{ and } w_i = 0, \\ -1 & \text{if } x_i = 1 \text{ and } w_i = 1. \end{cases} \tag{1}$$

Otherwise, the synaptic weight of the neuron remains unchanged for this presentation of the input pattern (see section 2 of *Methods* for a more detailed justification). We will refer in the following to the rule from Eq. 1 simply as BTSP rule.

One can further simplify this rule by deleting the dependence of weight updates on a drawing with a probability 0.5, and simultaneously reducing the probability of a plateau potential by this factor 0.5. Then the effective learning rate remains the same. We refer to this variant of Eq. 1 as the core BTSP rule. Note that core BTSP is still a probabilistic

rule, because its weight updates depend on randomly generated plateau signals. We show in Fig. S1 that core BTSP yields virtually the same results as the rule from Eq. 1. Hence, one could just as well use this even simpler rule. But we will base our theoretical analysis and numerical experiments on the stochastic rule from Eq. 1.

We demonstrate in Fig. 1D and E that the impact of a more complex rule for BTSP with continuous weight values, such as the rule proposed in (Milstein et al., 2021), can also be modelled by the simpler rule from Eq. 1 with binary synaptic weights. This holds if the pre- and postsynaptic neurons are connected by several synaptic release sites, whose individual strength can be approximated by binary values. One should note that most experimental data do not provide information about the number of synaptic release sites between two neurons, and changes in amplitudes of postsynaptic potentials at single release sites. Hence it remains open whether BTSP operates with continuously adapted synaptic strengths or with a small discrete number of synaptic strengths at single synaptic release sites.

Setup for evaluating the creation of memory traces through BTSP. Converging experimental data, see (Sugar and Moser, 2019) for a review, suggest that memory traces for experiences (episodes) are created in area CA1 of the hippocampus through the application of BTSP to synaptic connections from area CA3 to CA1. Hence the input neurons of our simple network model shown in Fig. 2A correspond to pyramidal cells in area CA3, and memory neurons to pyramidal cells in area CA1. Since lateral connections between pyramidal cells in area CA1 are rather rare, we model these memory neurons by a pool of n disconnected McCulloch-Pitts neurons. In other words, each memory neuron compares its weighted sum of synaptic inputs from the input neurons to a firing threshold. It outputs 1 (or "fires") if the sum exceeds the threshold, otherwise, it outputs 0.

We consider the benchmark task to create and recall memory traces for M input patterns $\boldsymbol{x}$, to which we will also refer as memory items in the following. Each input pattern is represented through the activity of m input neurons, see Fig. 2A. For simplicity, we assume that their activity can be characterized as firing or non-firing, i.e., by a single bit, so that each input pattern can be described by a bit vector of length m. Importantly, each of these memory items is presented just once to the network, while BTSP is applied to the synapses from the pool of m input neurons to the pool of n memory neurons. Performance for the benchmark task depends on various hyperparameters of the model. We focus on default values that are suggested by experimental data.

The parameter m corresponds to the number of pyramidal cells in area CA3, which is estimated to be around 0.25 million in the rat brain, and around 2.83 million in the human brain. The parameter n corresponds to the number of pyramidal cells in area CA1, which is estimated to be around 0.39 million in the rat brain, and around 14 million in the human brain. Since the large size of these values makes numerical simulations difficult, we have focused on values that are down-scaled by a factor of 1/10 from the size of these structures in the rat brain, and by a factor of 1/100 from the sizes in the human brain. Hence our default values in the model are $m = 25,000$ and $n = 39,000$.

Pyramidal cells of area CA1 are estimated to have around 30,000 dendritic spines in the mouse, see chapter 5 of (Andersen et al., 2007). Since some presynaptic neurons are likely to have synaptic connections on several spines, we assume in our model that a memory receives on average synaptic inputs from a random set of 15,000 input neurons. This suggests a connection probability of $f_w = 0.6$ for the randomly chosen connections from input neurons to memory neurons. The fraction f_p of bits that assume value 1 in an input pattern (corresponding to the fraction of neurons in CA3 that fire during the presentation of a memory item) was estimated by (Guzman et al., 2016) to have the value $f_p = 0.005$. Almost the same sparseness level was measured in a quite different brain area: Natural images were found to be encoded through the firing of a little less than 0.5% of neurons in the primary visual cortex of the monkey (Tang et al., 2018). Obviously, neural codes with such a high level of sparseness are also essential for achieving brain-like energy efficiency in neuromorphic hardware. Hence, we use $f_p = 0.005$ as default in our model for the fraction of 1's in a generic memory item. Note that in contrast to these biological data, most previous studies of memory models focused on neural codes for memory items with a much higher value of f_p. Since we are not aware of reliable experimental data on the firing threshold of pyramidal cells in CA1, we used grid search to set the

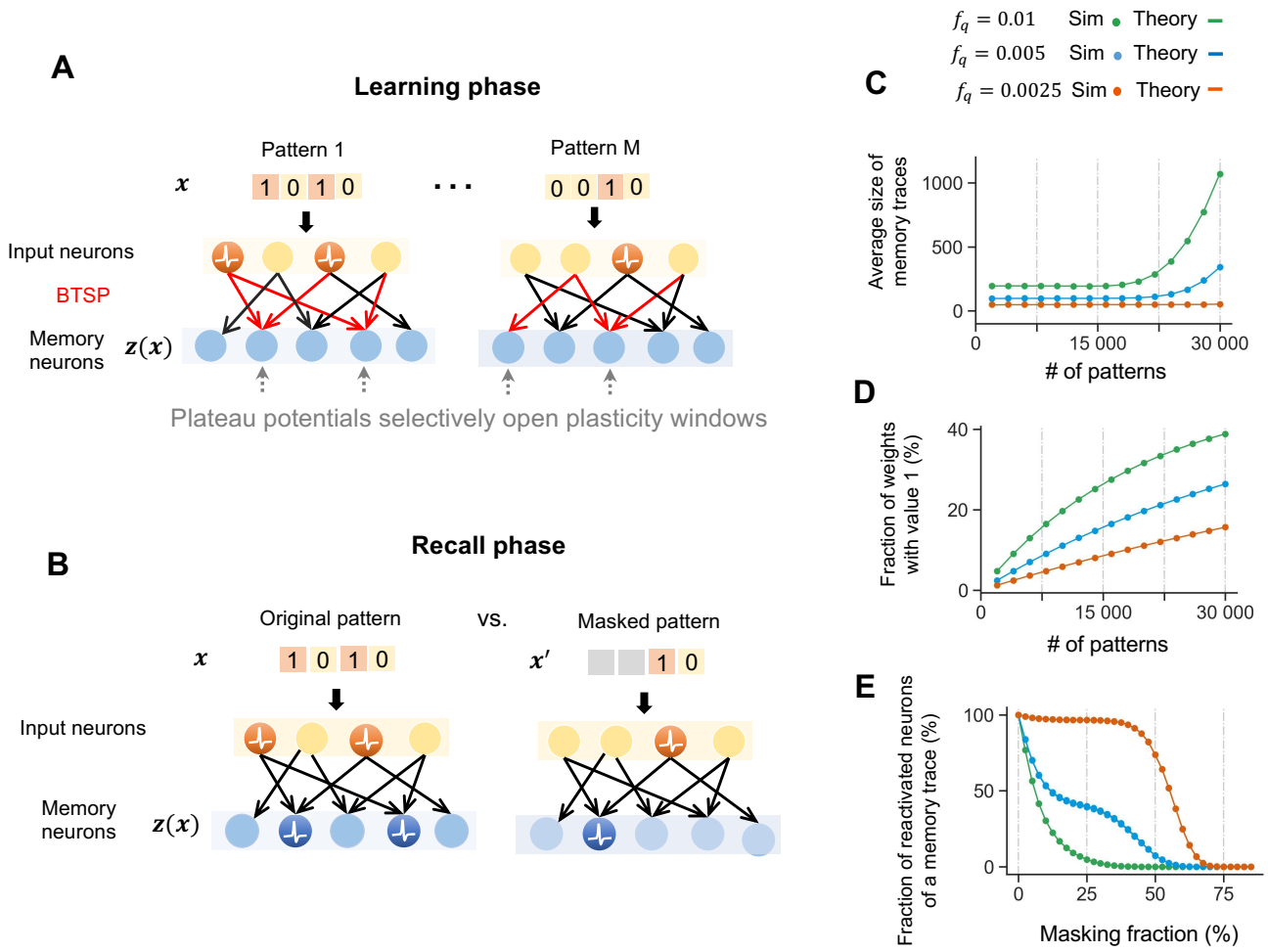

Figure 2. Structure and functional properties of memory traces created through BTSP. (**A**) Schematic illustration of the model for the creation of memory traces in the pool of memory neurons through BTSP. During the learning phase, synaptic connections between input neurons and memory neurons are subject to BTSP for a sequence of M memory items, with each item being presented once (one-shot learning). For each memory item, a random set of memory neurons receives a plateau potential, which opens for this memory item the plasticity window for synaptic connections from active input neurons to this memory neuron. (**B**) During the recall phase, when no further learning takes place, partially masked versions $\boldsymbol{x}'$ of previously learned memory items $\boldsymbol{x}$ are presented, where a random fraction of 1's in the input pattern is replaced by 0's. (**C**) The average size of memory traces (= number of memory neurons that fire during recall with the full memory item $\boldsymbol{x}$ as cue) depends somewhat on the total number M of memory items that have been learnt. It also depends on the probability f_q of a plateau potential in a memory neuron during learning. In particular, values of f_q that are substantially larger than the experimentally found value 0.005 lead to strongly increasing sizes of memory traces when many memory items are to be learnt. (**D**) The parameter f_q also impacts the fraction of strong synaptic weights after learning, especially for large numbers of memory items. (**E**) A larger value of f_q reduces the fraction of the memory trace that gets reactivated by a partial cue. One sees that the experimentally reported value 0.005 supports recall up to a fraction of one third masked 1's. Each of the panels C, D and E show that the results of numerical simulations agree very well with theoretical prediction. The latter were obtained from Eqs 13,16 and 20.

firing thresholds of CA1 neurons at a value that optimally supports recall from partial cues (see *Methods*).

After processing a sequence of M memory items $\boldsymbol{x}$ with one-shot learning through BTSP, a memory trace $\boldsymbol{z}(\boldsymbol{x})$ emerges for each memory item $\boldsymbol{x}$. It is defined by the set of memory neurons that fire after learning when input pattern $\boldsymbol{x}$ is presented. A key question is what happens after learning when a perturbed or partially masked version $\boldsymbol{x}'$ of one of the memory items $\boldsymbol{x}$ is presented to the network, while BTSP is no longer active, see Fig. 2B. We refer here to the fraction of 1's in an input pattern that are replaced by 0 as the masking fraction. Recall with such masked cues is motivated by the goal to recall conjunctive memories or episodes from just some of their components. We also considered recall with noisy perturbations of memory items where not only 1's are replaced by 0's but also 0's are replaced by 1's, , but so that the expected number of 1's remains the same as in the original input patterns. We show in Fig.S4 that this yields essentially the same results as when a corresponding number of bits are masked, i.e., when only 1's are replaced by 0's.

It is desirable that the set $\boldsymbol{z}(\boldsymbol{x}')$ of memory neurons that are activated by a masked or perturbed cue $\boldsymbol{x}'$ is similar to the memory trace $\boldsymbol{z}(\boldsymbol{x})$ for the original memory item $\boldsymbol{x}$. We measure their dissimilarity by the Hamming distance (HD) between them. We scale this HD by relating it to the average HD between memory traces for different memory items $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ from the original set of M memory items. We refer to the quotient of these two HD's as relative dissimilarity. The memory trace $\boldsymbol{z}(\boldsymbol{x}')$ for a perturbed or masked input pattern $\boldsymbol{x}'$ can only be seen as being similar to the memory trace $\boldsymbol{z}(\boldsymbol{x})$ for the unperturbed memory item $\boldsymbol{x}$ if this relative dissimilarity is substantially smaller than 0.5.

Note that both HD values that occur in this measure need to be analyzed after all input patterns in the sequence have been learned. This is essential since these HD values change in general during the processing of the sequence of the M memory items through BTSP. In particular, one needs to take into account that weights which have been increased when a memory item $\boldsymbol{x}$ was processed through BTSP may become later subject to LTD when a subsequent item in the list of M memory items has common 1's and activates also some of the same memory neurons, provided that these receive again a plateau potential when the subsequent memory item is processed. Therefore one cannot use theoretical methods that have previously been developed for analyzing memory traces in HFNs, such as those in (Hopfield, 1982, 2007; Widrich et al., 2020), where the order of memory items during learning was irrelevant because the weight change did not depend on the prior value of the weight. Therefore new tools from combinatorial mathematics and probability theory were needed to derive a theory for memory systems created by BTSP. The curves in Fig. 2C-E show that the resulting theory is very successful in predicting the outcome of numerical simulations.

Functional role of the probability f_q of the stochastic gating signal. Virtually all models for memory in neural networks are based on the postulate of Hebb: Synapses between neurons that fire together are strengthened. BTSP provides a radical departure from this postulate, and also from STDP, since BTSP does not depend on the firing of the postsynaptic neuron. Instead, it depends on presynaptic activity, the current value of the synaptic weight, and the current presence or absence of a plateau potential in the postsynaptic neuron. The experimental data of (Grienberger and Magee, 2022) suggest that one can model the presence of plateau potentials in CA1 neurons by a stochastic rule, whereby CA1 neurons receive during learning independently with probability f_q of plateau potential. We show that the value of this parameter f_q, which has no analogue in previous memory models, influences significantly the properties of the resulting memory system.

We show in Fig. 2C and D that a larger value of f_q increases the size of memory traces, i.e. the number of memory neurons that fire when a memory item is presented. But it also increases the overlap of memory traces for different memory items, which is in general disadvantageous. Furthermore, Fig. 2E shows that a larger value 0.01 of f_q reduces the capability to recall a memory trace with a partial memory item where a fraction of the 1's in the input vector have been masked, i.e., set to 0. This is also partially caused by the fact that plateau potentials not only enable LTP, but also LTD; the latter in case that the current weight is already large. Therefore a larger value of f_q implies

that more weights that were first set to 1 for one item in the list of M memory items become subject to LTD during the application of BTSP to subsequent memory items in the list. This tends to reduce the weighted sum that arrives at each neuron in the memory trace of a memory item. If in addition some 1's in the input pattern are set to 0, the resulting weighted sum may no longer reach the firing threshold of the memory neuron. On the other hand, Fig. 2E also shows that the relatively low value 0.005 for f_q that is suggested by the experimental data of (Grienberger and Magee, 2022) produces substantially more robust memory traces that support reliable recall even when a third of the bits of the memory item have been masked.

We will later show in Fig. S4 and S5 that an even lower value of f_q provides further performance improvement for recall with partial cues and input completion. On the other hand we will show in Fig. 6C that a lower value of f_q reduces the repulsion effect.

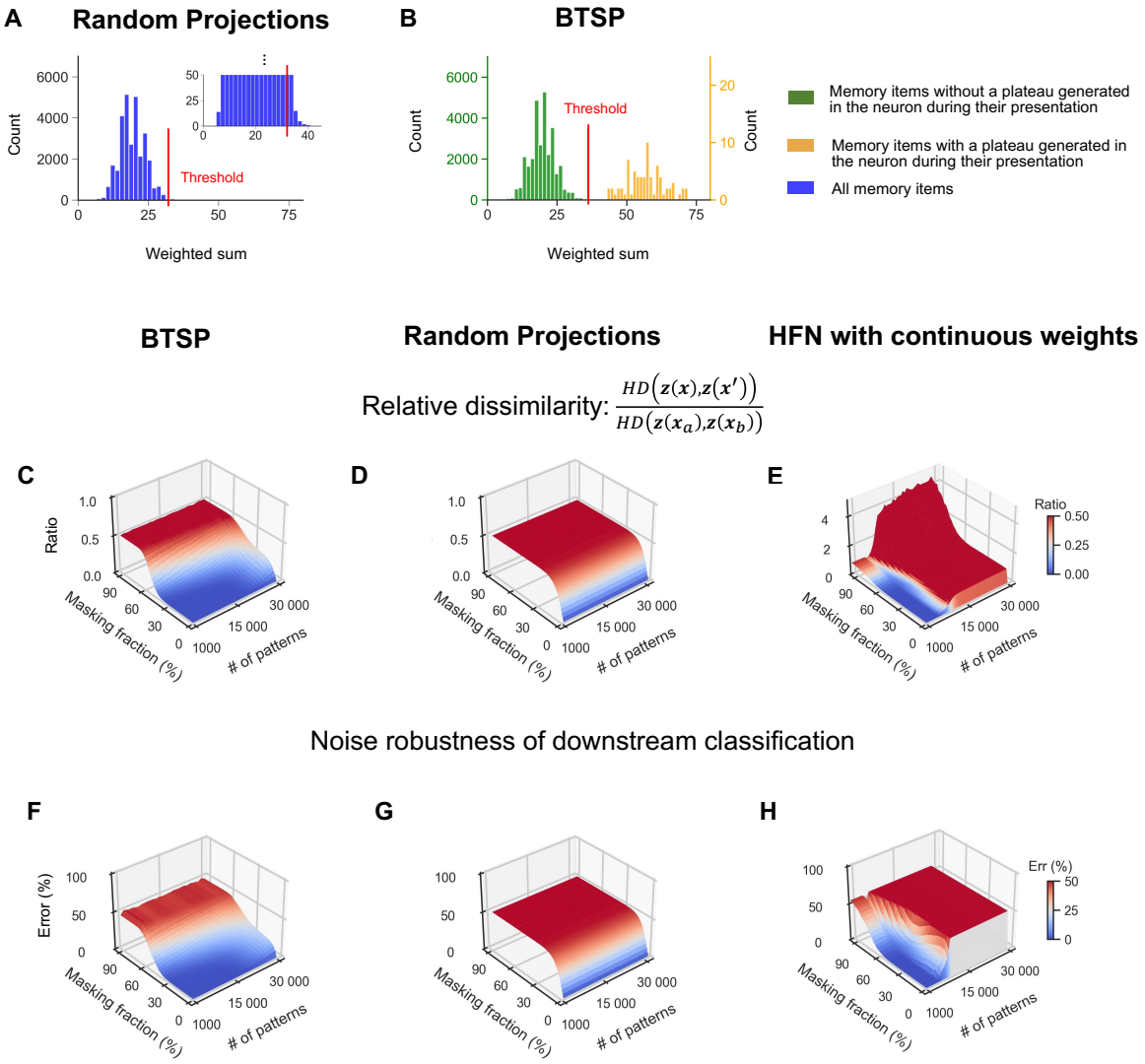

Figure 3. Recall with partial cues for memory traces created via BTSP, a random projection (RP), and in a classical Hopfield network (HFN) with continuous weights. (**A**) The distribution of synaptic input sums to a sample memory neuron for all 30,000 memory items for the case of a RP. The red line marks the optimal value of the neuron threshold for minimizing the relative dissimilarity for a masking fraction of 33%. One sees both in the full-scale plot and in the inset that moving the threshold to a lower value would result in the inclusion of this neuron in a significantly larger number of memory traces, thereby increasing their overlap. (**B**) The distribution of synaptic input sums to a sample memory neuron for all 30,000 memory items that occur during learning with BTSP. The memory items for which a plateau potential occurred during learning cause significantly larger weighted sums during recall (yellow histogram). The histogram for the other memory items is shown in green. Therefore, it is easy to choose a value of the firing threshold that maintains a significant distance from both clusters of weighted sums. In this way, the neuron becomes part of the memory trace for all memory items for which a plateau potential occurred during learning, and the weighted sum stays for these memory items above the threshold even when a substantial fraction of 1's in the memory items have been masked. Panels (**C**) and (**D**) show that the relative dissimilarity between the original memory trace and the memory trace recalled by a partial cue is significantly smaller for BTSP than for a RP, due to the mechanism explained in panels A and B. Furthermore, they show that the relative dissimilarity remains in the case of BTSP small for masking fractions up to 33%, whereas for a RP it approaches already for small masking fractions the value 0.5, where one can no longer distinguish to which memory item the trace for the masked cue belongs. (**E**) shows corresponding results for a classical HFN. Despite the substantially larger information content associated with its continuous weights, the observed memory capacity is not significantly larger than that of the memory systems with binary synaptic weights created by BTSP. Note that we consider here only the case of energy-efficient sparse representations of memory items, as they occur in area CA3. Panels (**F**), (**G**) and (**H**) exhibit similar performance differences for generic downstream classifications of memory traces. Shown are the results of simulations, averaged over 5 runs with new random generation of memory items and plateau potentials.

BTSP creates a system of memory traces for sparsely encoded memory items that supports recall with partial cues. The fact that BTSP is gated by a stochastic component, random plateau potentials, endows the memory systems that it produces with generic advantages of stochastic memory allocation: The memory space, i.e. the population of memory neurons, is used uniformly, and artefacts that might result from a particular structure of the ensemble of input patterns are reduced. Random projection and random hashing are commonly used strategies for stochastic memory allocation in digital computers, see (Dasgupta et al., 2017) for a review and references. A random projection (RP) is a linear map from input neurons to memory neurons that is defined by a stochastic matrix of 0's and 1's, with subsequent thresholding of output values. In order to facilitate a comparison, we use here the same fraction of 1's in the matrix of RPs that result from learning in the weight matrix for synaptic connections from input neurons to memory neurons in the case of BTSP.

A key difference between BTSP and RPs is that the projecting matrix for RPs is chosen independently from the memory items that are to be stored, whereas BTSP generates synaptic weights for a specific set of memory items. We demonstrate in Fig. 3 that this provides a substantial functional advantage to BTSP with regard to the recall of memory items with masked cues. The reason for this functional advantage is elucidated in panels A and B of Fig. 3. No matter how one chooses the threshold for binarizing the outputs of a RP, if one wants to have that the weighted sum is for a given number of memory items above this threshold, the weighted sum will be for many of these memory items just above the threshold, see Fig. 3A . Hence masking of even a small fraction of 1's of in these memory items lets the weighted sum drop below the threshold, i.e., the memory neuron is not activated by the masked input pattern.

In contrast, the distribution of weighted sums that result from BTSP in a generic memory neuron is bi-modal, as shown in Fig. 3B: One cluster consists of weighted sums for memory items for which a plateau potential occurred in this neuron when the item was learnt, and the other cluster consists of memory items for which this did not happen. Further examples for this bimodal distribution are shown in Fig. S2 for other memory neurons. An obvious advantage of this bimodal distribution is that one can set the firing threshold of a memory neuron between these two clusters. Then a perturbation of a small fraction of bits of a memory item is not likely to move the weighted sum to the other side of the threshold. This is confirmed by an evaluation for large sets of memory items in Fig. 3 C and D: Recall with cues where up to a third of the input bits are masked works well for memory traces that are created by BTSP, but RP tolerates hardly any masking of input bits (even though we optimized its threshold for producing binary outputs for recall with masked cues, see *Methods*). These results of numerical simulations are corroborated by theoretical predictions that are depicted in Fig. S3. Simultaneously one sees there that the results of numerical simulations and theoretical predictions agree very well.

HFNs with continuous weights are arguably the most commonly considered CAM model. They require weights that can assume a large number of different values because their standard learning rule for storing memory items from (Hopfield, 1982; Tsodyks and Feigel'man, 1988) requires an increasing number of values for synaptic weights when more memory items are stored. In fact, the required number of weight values grows linear with the number of stored memory items. We are not aware of other plasticity rules that install a functionally powerful CAM in HFNs with any number of weight values that is much smaller than the number of stored memory items. In particular, there is apparently no learning rule that installs a CAM in HFNs with binary weights. Furthermore, the results of (Tarkov, 2019) show that if one first trains a HFN with a learning rule that employs continuous weights, such as the rule of (Tsodyks and Feigel'man, 1988), and then quantizes the synaptic weights, the memory capacity of the HFN drops drastically.

Because of the much larger information content of synaptic weights in a HFN with continuous weights, it is inadequate to compare the capacity of a memory system with binary weights that is created through BTSP with that of a HFN with continuous weights. Nevertheless, we have carried out such a comparison in Fig.3 C and E, and also in F and H for another task. One sees that the performance of memory systems with binary weights that are created through BTSP is close, and in some cases even superior to the memory system with continuous weights of a

classical HFN. We used here for the HFN the same connection probability $f_w = 0.6$ between neurons as for BTSP, and also the same input patterns. For the HFN we write $\boldsymbol{z}(\boldsymbol{x})$ for the set of neurons that are activated by a cue $\boldsymbol{x}$ after 100 synchronous update steps of the HFN after the presentation of the cue.

We show in Fig. S6B theoretical predictions for a scaled-up setting for BTSP where we consider 100 times larger input patterns, that correspond roughly to the size of area CA3 in the human brain, i.e., to the size of memory items that are represented through the firing of neurons in CA3 that project to CA1. We simultaneously reduced there the fraction of input bits with value 1 by the factor 1/100, so that on average the same number of input bits have value 1 as in our standard setting. These theoretical predictions suggest excellent recall capability even if up to a 2/3 fraction of a memory item is masked in a cue, for up to 800,000 memory items.

A direct functional consequence of the robust recall that is achieved by memory systems created with BTSP for masked input patterns is demonstrated in Fig.3 F - G. A key question for downstream classifications of memory items to memory traces is how robust these classifications are to perturbations of an input pattern $\boldsymbol{x}$. We show in Fig.3 F that the noise reduction that BTSP achieves in the mapping of memory items to memory traces translates directly into the high robustness of a generic downstream classification. We considered here a generic downstream classification of memory items that is defined by a thresholded linear readout that is applied to the output layer of the memory network, i.e., to the memory traces, with randomly chosen integer readout weights, uniformly sampled from the range -8 to +8, and a fixed threshold 0 of the readout neuron. Robustness was measured in terms of the fraction of memory items $\boldsymbol{x}$ for which masking of the memory item produced an error, i.e., changed the classification result. One sees in Fig.3 F - G that such a downstream classification was substantially more robust when memory traces were created through BTSP rather than by a RP. The performance of the HFN with continuous weights was again not superior to that of the memory system with binary weights created by BTSP.

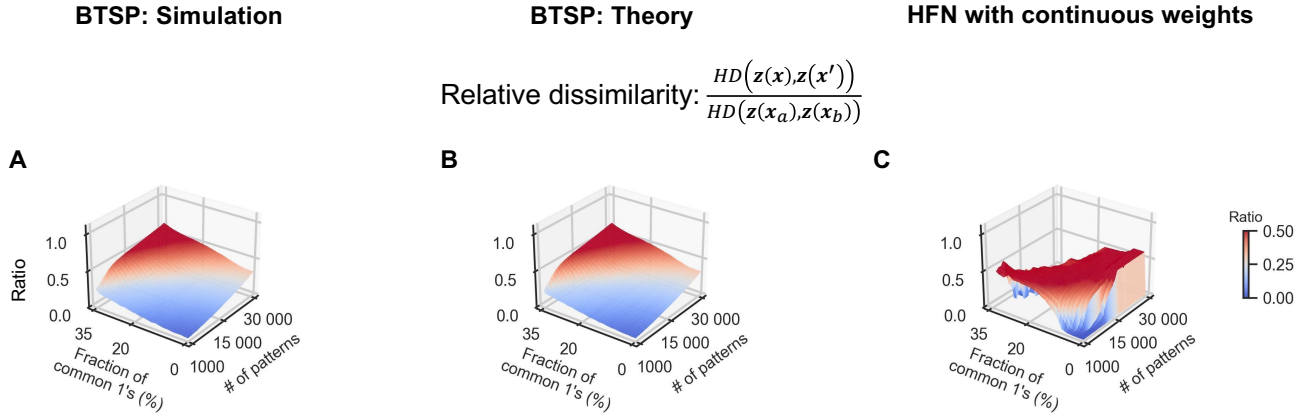

Figure 4. Recall with partial cues from memory traces created by BTSP and HFN for non-orthogonal memory items. Relative dissimilarity of memory traces for non-orthogonal memory items, analysed through numerical simulations (**A**) and theory (**B**) for BTSP, as well as through numerical simulations for the HFN with continuous weights (**C**). The partial cues $\boldsymbol{x}'$ in the relative dissimilarity measure refer to patterns with 33% of elements masked. A comparison of panels A and B shows that the theory for BTSP predicts the results of numerical simulations very well. Grid search was used to optimize the firing threshold of neurons (see *Methods*). Average results of 5 trials are depicted in panels A and C.

Memory traces for non-orthogonal memory items. Episodic and conjunctive memory require the storage of memory items that may have common components, and are therefore not orthogonal. In contrast, studies of HFNs usually assume that the stored memory items are orthogonal. Hence the question arises whether BTSP is able to create functionally useful memory traces also for non-orthogonal memory items, in particular for memory items

that have common components or features, i.e., common 1's. Consequently we measured the non-orthogonality of memory items by the average fraction of common 1's of any two memory items in the list of M items. For independent memory items the expected number of common 1's is 0.625, i.e., 0.5% of their expected number of 1's.

We show in Fig. 4A through numerical simulations that BTSP also creates for non-orthogonal memory items a memory system that supports recall with incomplete cues, where 33% of the bits are masked. Whereas most theoretical studies of memory models are only valid for orthogonal memory items, we show in Fig. 4B that the theory for BTSP that we have developed, in particular Eqs.29 and 30, also works for non-orthogonal memory items, and produces very similar results as the numerical simulations.

Both simulations and theory show that for BTSP a fraction of 10 to 20% common 1's, i.e., up to 40 times more than for independent memory items, only mildly degrades the quality of recall with partial cues. In contrast, Fig. 4C shows that even a HFN with continuous weights cannot match the performance of BTSP for storage and recall of non-orthogonal memory items with binary weights.

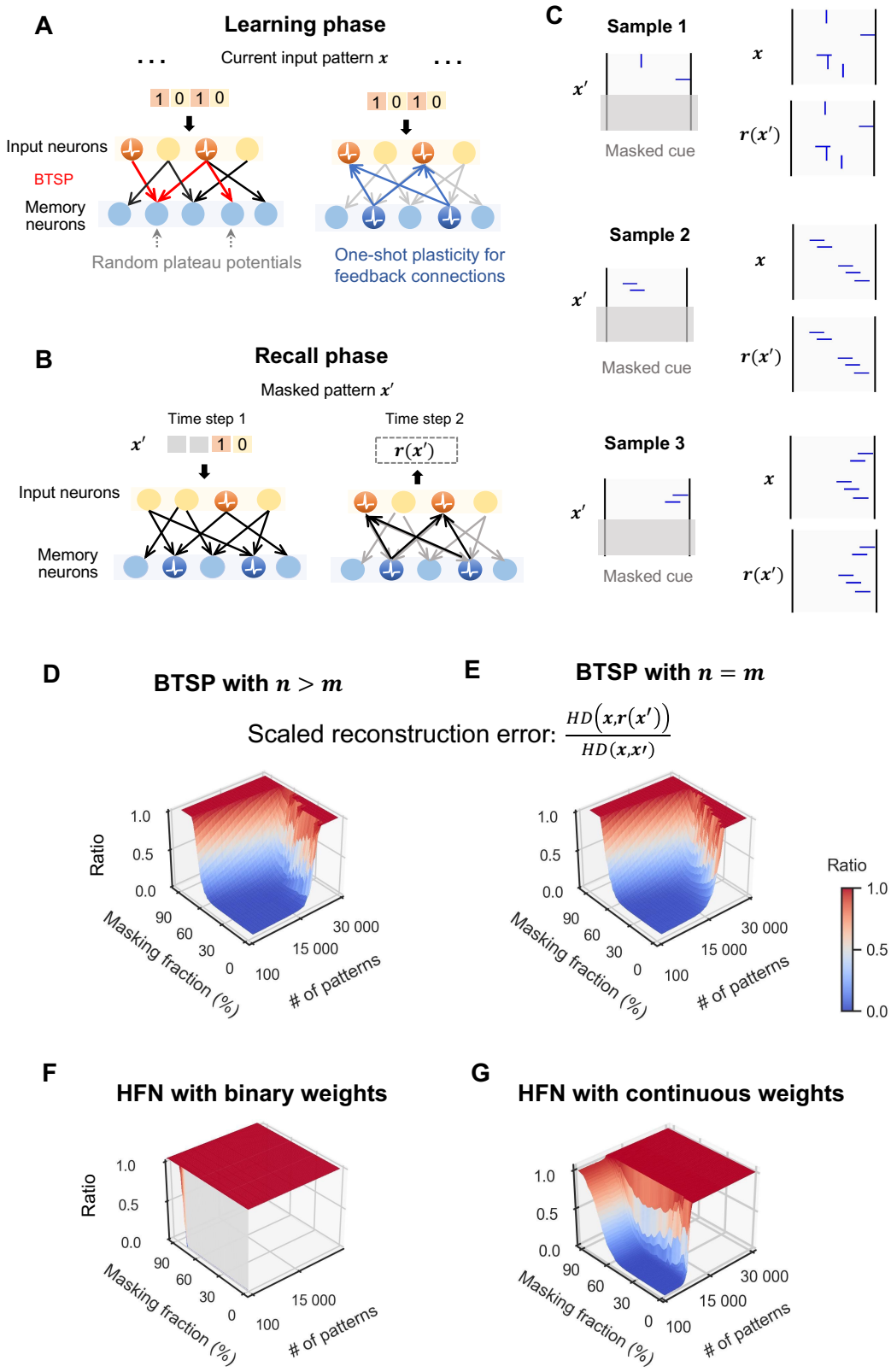

Figure 5. *caption next page*

Figure 5 (*previous page*). BTSP enables input reconstruction when feedback connections are added. (**A**) Illustration of the model during the learning phase. For each memory item $\boldsymbol{x}$ the forward connections are trained through one-shot learning via BTSP in a single step, as before. Before the next memory item is processed, the feedback connections are trained in a single step through Hebbian-like plasticity (see *Methods* for details). (**B**) Illustration of input reconstruction in this model after learning with BTSP. Input neurons encode a partially masked cue $\boldsymbol{x}'$ at one step, and are overwritten by the internally generated pattern $\boldsymbol{r}(\boldsymbol{x}')$ at the next time step. (**C**) Demonstration of input completion via BTSP for an ensemble of 2,000 input patterns that are easy to visualize: The patterns were collections of 5 line segments (each 8 pixels long) in horizontal or vertical direction, in random positions. A generic pattern of this type is shown as the first plot. During the recall phase, the last 20 rows of each original image were masked (indicated by the grey area). To make input completion more challenging, we included in the list of 2,000 memory items also a pair of patterns that agreed in 3 of the 5 line segments, see the 2nd and 3rd plot. One sees that BTSP with feedback connections effectively distinguishes and accurately reconstructs also masked versions of these two very similar memory items. (**D**) shows a quantitative analysis of the input reconstruction capability of the BTSP model for the same ensemble of random input patterns as used in Fig. 3. We used the HD ratio $HD(r(\boldsymbol{x}'),\boldsymbol{x})/HD(\boldsymbol{x},\boldsymbol{x}')$ for a memory item $\boldsymbol{x}$ that occurred during the learning phase and the internally generated pattern $\boldsymbol{r}(\boldsymbol{x}')$ for a masked version $\boldsymbol{x}'$ of this memory item as a scaled measure for the input reconstruction error. A small value of this quotient indicates a small reconstruction error, whereas values 1 and larger indicate that no input reconstruction has been achieved. (**E**) shows the input reconstruction capability for a minor variation of this model where the number n of neurons in the memory later was reduced from 39,000 to 25,000, which is the number of neurons in the input layer, and therefore also the bit length of a memory item. The performance is only affected very little by this reduction. Its performance should be compared with the input reconstruction capability of a HFN of the same size 25,000 with binary weights, shown in panel (**F**), and with continuous weights in panel (**G**) for the same ensembles of memory items. One sees that the binarization of HFNs significantly reduces their input reconstruction capability for sparse input patterns and BTSP model exhibits competitive input reconstruction capability compared to the HFN with continuous weights.

BTSP endows memory traces with attractor properties that are beyond the reach of RPs. We had already shown in Fig. 3 that memory traces that are created through BTSP have an inherent attractor property that cannot be achieved by RPs: Recall with partial cues $\boldsymbol{x}'$, where a significant fraction of the 1's in a stored memory item $\boldsymbol{x}$ are masked, activates a memory trace $\boldsymbol{z}(\boldsymbol{x}')$ whose HD to $\boldsymbol{z}(\boldsymbol{x})$ is in general substantially smaller than the HD between $\boldsymbol{x}$ and $\boldsymbol{x}'$. In Fig. S3A we show that this also holds when the recall cue $\boldsymbol{x}'$ is a randomly perturbed version of $\boldsymbol{x}$, where not only 1's are replaced by 0's but also 0's are replaced by 1's. In contrast, memory traces created through a RP have no such attractor property according to Fig. 3D.

We are addressing here the question whether this attractor quality that is induced by BTSP can also produce input reconstruction, i.e., the characteristic property of a CAM. To test that, one needs to add a generative component to the model, i.e. synaptic connections from the output layer back to the input layer. These feedback connections mimic synaptic connections in the brain that enable the reconstruction of memory content in the neocortex from a memory trace in area CA1. Since the biological implementation of these feedback connections and their synaptic plasticity remain opaque, we have chosen here the simplest possible model: Random feedback connections from the memory neurons to the input neuron layer, see the right parts of panels A and B of Fig.5, that are subject to a one-shot Hebbian synaptic plasticity rule during the learning phase (see *Methods*). The neurons in the input layer are modelled like the neurons in the output layer as McCulloch-Pitts neurons. The resulting input reconstruction capability after one-shot learning with BTSP and the Hebbian plasticity is illustrated in Fig. 5C. Input completion for a cue $\boldsymbol{x}'$ is produced immediately at the next time step after the presentation of a cue $\boldsymbol{x}'$: The cue $\boldsymbol{x}'$ is then removed, and the input neurons are driven by feedback from the output layer. An input neuron now assumes the value 1 if its weighted sum of inputs from the output layer exceeds its firing threshold. We write $\boldsymbol{r}(\boldsymbol{x}')$ for the bit vector that results in this way from the input cue $\boldsymbol{x}'$ at the preceding time step. If the recall cue $\boldsymbol{x}'$ is a masked version of a memory item $\boldsymbol{x}$ that occurred during learning, the internally generated pattern $\boldsymbol{r}(\boldsymbol{x}')$ approximates in general the

original memory item $\boldsymbol{x}$ very well, see the first panel of Fig. 5C.

We analyze this input reconstruction capability quantitatively in Fig. 5D - G, for the same sparse input patterns as in Fig.3. We see in Fig. 5D and E that the $HD(\boldsymbol{x}, \boldsymbol{r}(\boldsymbol{x}'))$ between an original memory item $\boldsymbol{x}$ and the internally generated completion $\boldsymbol{r}(\boldsymbol{x}')$ of a masked version $\boldsymbol{x}'$ of the memory item is in general substantially smaller than the HD between $\boldsymbol{x}$ and $\boldsymbol{x}'$, even when large numbers of memory items $\boldsymbol{x}$ are stored, and if a large fraction of its bits is masked in the recall cue $\boldsymbol{x}'$. In order to facilitate a performance comparison with HFNs, we first show in Fig. 5E that BTSP has basically the same input reconstruction performance when we reduce the size n of the output layer to the size $m = 25,000$ of the input layer, i.e., to the bit length of the input patterns. We then show in Fig. 5F that a HFN of size 25,000 with binary weights, resulting from rounding after learning with continuous weights according to (Tsodyks and Feigel'man, 1988), is not able to carry out input completion for more than 100 stored memory items. Other strategies for binarizing the weights of a HFN yield the same performance according to Fig. S7. This drastic drop in memory capacity after rounding of weights had already been demonstrated in (Tarkov, 2019) for the case of densely coded memory items. Finally, we show in Fig. 5G the input reconstruction performance of a HFN of size 25,000 with continuous weights. One observes that the CAM capability of the memory system created by BTSP is as competitive as that of an equally sized HFN with continuous weights.

This result provides good news for the improvement of CAMs in neuromorphic hardware. CAM models had already been implemented through crossbar arrays of memristors in neuromorphic hardware, see (Hu et al., 2015; Sun et al., 2021; Wang et al., 2020), using off-chip learning of synaptic weights for a HFN with high weight resolution. A comparison of panels E and G of Fig. 5 suggests that at least the same CAM quality can be reproduced by crossbar arrays of memristors with just 2 resistance states, which can be set through on-chip learning via BTSP. In addition, BTSP enables instant memory recall, whereas the HFN model requires multiple iterations for convergence to an attractor and, therefore, takes more time for the production of a completed input pattern. We show in Fig.S5 that the input completion performance with BTSP becomes even better for a lower value of f_q.

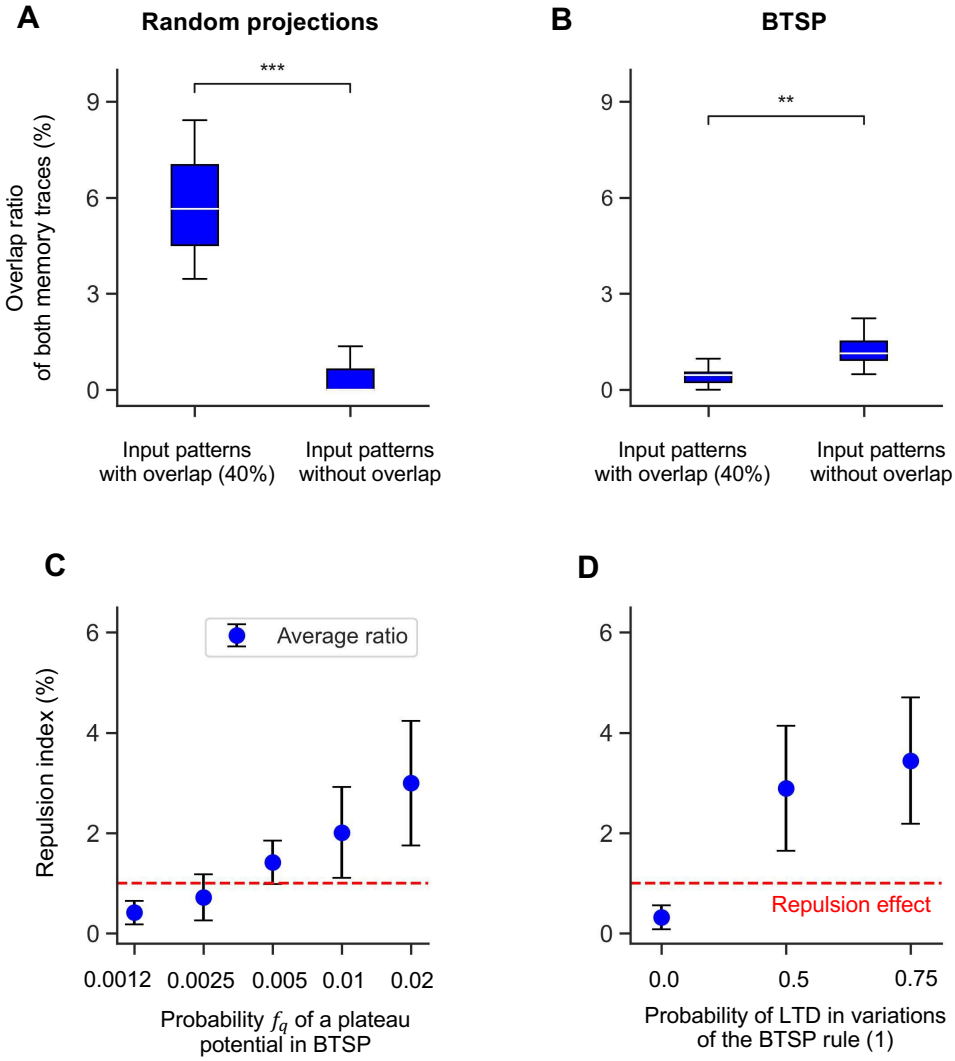

Figure 6. In contrast to random projections, the BTSP rule can reproduce the repulsion effect of human memory. (**A**) The overlap ratio of the memory traces created by a RP for a pair of memory items with 40% overlap is shown on the left, and for a pair of unrelated memory items on the right. The overlap ratio is higher for the more similar memory items ($***p < 0.001$, two-tailed one-sample t-test). (**B**) The same analysis was carried out for memory traces that resulted from BTSP. Here the overlap ratio is smaller for more similar memory items ($**p = 0.002 < 0.01$, two-tailed one-sample t-test), like in corresponding experimental data from the human brain (Chanales et al., 2017). Details of the data generation are provided in the *Methods*; $f_q = 0.02$ was used in panel B. Panels (**C**) and (**D**) elucidate the impact of the plateau potentials probability f_q and of the LTD probability (default value 0.5) on the repulsion effect. The strength of repulsion was measured by the repulsion index, which is obtained by dividing the overlap ratio of memory traces for a pair of unrelated memory items by the overlap ratio for a pair of memory items with 40% overlap. Values above the red dashed lines that are defined by value 1 indicate the occurrence of the repulsion effect. One sees that values for f_q that are below our biologically derived default value 0.005 do not produce repulsion, and also that the LTD component of BTSP is essential for repulsion. In fact, the repulsion effect becomes even stronger when the LTD probability is increased beyond its default value. The error bars in both panels represent the standard deviation of the overlap ratio over 20 trials with new generations of random connection weight, input patterns, and plateau potentials.

BTSP induces a repulsion effect for closely related memory items. Curiously, BTSP induces for similar memory items an effect that is seemingly incompatible with the previously described attractor properties: Memory traces for similar memory items are pulled away from each other. A closer look shows that this repulsion effect is, in fact, compatible with the previously demonstrated attractor properties of BTSP-induced memory traces: The difference is that these similar memory items occur during learning, not only during subsequent recall. Specifically, we show that the LTD part of BTSP makes the memory trace dissimilar to that of a previously processed similar memory item. So far the LTD part of BTSP was always running in the background and gave rise to the desirable effect that the number of strong synaptic connections remained limited (see Fig. 2D). Here we elucidate another important functional impact of the LTD part of BTSP.

An obvious functional advantage of this repulsion effect is that it enables differential downstream processing of memory items that differ in just a few features. Arguably, this is a key trait of higher cognitive function. In fact, experimental studies of memory traces in the human brain, especially in area CA1, show that the human brain exhibits a repulsion effect (Chanales et al., 2017; Wammes et al., 2022; Wanjia et al., 2021; Zheng et al., 2021). However, previous memory models have not been able to reproduce this. In particular, a RP produces an opposite effect, whereby memory traces for more similar memory items have more similar memory traces, see Fig. 6A. To the best of our knowledge, BTSP provides the first learning-based model for the repulsion effect of human memory.

Fig. 6B shows results of numerical simulations for an experiment with BTSP that mimics the experimental design of studies of the repulsion effect in the human brain (Chanales et al., 2017; Wammes et al., 2022; Wanjia et al., 2021; Zheng et al., 2021): A sequence of M memory items is learned through BTSP, where these memory items are pairwise unrelated except for two that have a large overlap ratio of 40%, but may appear at any position in the sequence. We evaluate the similarity of resulting memory traces similarly as in these experiments, where the Pearson correlation coefficient was applied to voxels of fMRI recordings: We measure the overlap ratio of two memory traces by their fraction of common 1's, divided by the average number of 1's in a memory trace. Also the similarity (overlap) of two memory items is measured through the fraction of common 1's. Whereas no repulsion effect occurs with RPs according to Fig.6A, a significant repulsion effect is induced according to Fig. 6B by BTSP. Note that its strength is in this model comparable to that in the human brain, see Fig. 3C of (Chanales et al., 2017).

One can, in fact, clearly understand how the LTD part of BTSP induces this repulsion effect. If a memory neuron receives during learning plateau potentials for both of two similar memory items, the strength of most of its synaptic connections from neurons that are active in both memory items will have a value 0 after learning: These synaptic weights are first increased and then decreased by the LTD part of BTSP. This effect reduces the overlap of the resulting memory traces because this memory neuron will not belong to the memory trace for either of the two similar memory items. On the other hand, repeated plateau potentials to the same memory neuron for two memory items that have largely non-overlapping sets of 1's will only reduce weights from those few input neurons that assume value 1 for both input patterns. Therefore, this memory neuron is more likely to fire for both memory items after learning has been completed, thereby increasing the overlap ratio of the resulting memory traces (see the right part of Fig. 6B).

Discussion

We have created and examined a simple rule for BTSP, the recently discovered one-shot learning rule in the brain (Bittner et al., 2017; Magee and Grienberger, 2020; Milstein et al., 2021), see Fig. 1. More detailed BTSP data and rules with continuous weights can be reproduced by this simple rule when multiple synaptic connections are taken into account, see Fig.1 C and D. Importantly, this BTSP rule is sufficiently simple so that its functional impact can be studied analytically. We have shown that the resulting predictions agree very well with the results of numerical simulations.

Whereas previous models for BTSP aimed at reproducing experimental data from mice during navigation in mazes (Cone and Shouval, 2021; Li and Roxin, 2023), we have focused here on functional implications of the simple BTSP rule for the creation of memory systems for large numbers of generic sparsely encoded memory items. One of the most surprising and intriguing features of BTSP is that it does not rely on postsynaptic neural activity, like virtually all other synaptic plasticity rules that have been studied in the context of memory models or elsewhere in theoretical neuroscience. Instead, it relies on external stochastic gating signals that open for several seconds the gate for synaptic plasticity in a neuron. We have elucidated the functional impact of the probability of the occurrence of these gating signals, the parameter f_q, both theoretically and through numerical simulations, see Fig. 2 C, D, E and Fig. 6 C. In particular, we have shown that the biologically found value $f_q = 0.005$ can be seen as a sweet spot for combining several functionally attractive features of BTSP. We have also shown that smaller or larger values of f_q can enhance selected functional properties of BTSP, see Figs. S4 and S5.

On a more abstract level, our results show that this stochastic feature of BTSP can be seen as a mechanism for porting functionally attractive properties of the fly algorithm (Dasgupta et al., 2017) into the realm of synaptic plasticity and learning. More precisely, BTSP embeds these advantages of stochasticity into a learning mechanism that tailors the resulting "random projection" to a particular ensemble of memory items that are to be stored. We have shown that this makes BTSP superior to the fly algorithm and random projections as methods for neural coding because BTSP induces powerful attractor properties for the memory traces that it creates, see Figs. 3, 4, 5. In fact, to the best of our knowledge, the simple BTSP rule is the first learning rule that is able to create through one-shot learning a CAM with binary synaptic weights. Hence the creation of memory systems through BTSP provides a viable alternative to HFNs, i.e., to the most studied model for CAM in neural networks. HFNs require continuously valued synaptic weights, or at least synaptic weights that can assume many more distinguishable values than biological synapses (Bartol Jr et al., 2015). Their large number of weight values also makes it cumbersome to implement them with now available large crossbar arrays of memristors of hardware for in-memory computing. Therefore previous neuromorphic implementations of CAM could only address small CAM instances (Hu et al., 2015; Sun et al., 2021; Wang et al., 2020), and could not combine them with on-chip learning.

Furthermore, in contrast to HFNs, the CAMs that BTSP creates are not restricted to independent or orthogonal memory items. Hence this new type of CAM can also be used for storing episodic memories or conjunctive memories that share common components. Furthermore, we show in Fig. 6 that BTSP reproduces the repulsion effect of the human brain that pulls memory traces for similar memory items apart, as documented in (Chanales et al., 2017; Wammes et al., 2022; Wanjia et al., 2021; Zheng et al., 2021). This effect can be seen as a vital component of higher cognitive function since it enables differential processing of experiences that differ only in few but possibly salient features. However, the repulsion effect could not be reproduced with previously considered learning rules.

Thus, altogether we have shown that a simplified version of the biological synaptic plasticity rule BTSP throws new light on the design and nature of CAM in the brain, and that it paves the way for porting essential parts of its functional benefits into large-scale energy-efficient hardware with on-chip learning capability.

Table of content for Methods

Table 1: Definitions and explanations of key parameters used in the theoretical analysis. 21

A learning theory for BTSP 21

Background and preparations for memory trace estimates 21

Estimates of properties of memory traces for randomly drawn memory items 21

Estimates of firing probability of memory neurons after learning M randomly drawn memory items . . 21

Estimates of expected HD between memory traces for the original and masked memory items 24

Estimates of expected HD between memory traces for different randomly drawn memory items 26

Estimates of properties of memory traces for non-orthogonal memory items 26

Estimates of firing probability of memory neurons after learning M non-orthogonal memory items . . 26

Estimates of expected HD between memory traces for the original and masked memory items 28

Estimates of expected HD between memory traces for different non-orthogonal memory items 29

Details of experimental setup and simulations 30

Detailed justification of the stochastic aspect of the BTSP rule of Eq. 1. 30

Details of simulation of continuous weight changes by BTSP rule (Fig. 1D and E) 30

Details of analysis of the masking fraction (Fig. 2) 30

Details of comparison with other memory models (Fig. 3). 31

Details of analysis of the masking fraction for non-orthogonal memory items (Fig. 4) 31

Details of experimental setup for the input completion (Fig. 5) 31

Details of analysis of the repulsion effect (Fig. 6) 32

Methods

1 A learning theory for BTSP

1.1 Background and preparations for memory trace estimates

For simplicity, our theoretical analysis will be presented using the BTSP rule from Eq. 1, under the assumption that the weight update is made independently for each synapse with a probability of 0.5. This methodological assumption aims to simplify the analytical framework and improve the comprehensibility of the results, without compromising the integrity of our findings. As substantiated by the numerical simulations and theoretical predictions presented in Fig.S1B and C, this approach demonstrates consistency with the broader results.

We will employ the properties of the binomial distribution for the memory trace estimates.Key parameters used in the following derivations are summarized in Table 1. Additionally, we introduce the following lemma crucial for handling the parity of weight updates in Eqs.12, 13, 22 and 23.

Lemma 1. *Let $f(k_x, N_x, p_x)$ be the probability of binomial distribution that a random variable X achieves k_x success in N_x Bernoulli trials. Then, the probability that X is even can be given by*

$$P(X \text{ is even}) = \frac{1}{2}\left(1 + (1-2p_x)^{N_x}\right). \tag{2}$$

Similarly, the probability that X is odd can be given by

$$P(X \text{ is odd}) = \frac{1}{2}\left(1 - (1-2p_x)^{N_x}\right). \tag{3}$$

Proof. Note that

$$((1-p_x)+p_x)^{N_x} = \sum_{k_b=0}^{N_x} C_{N_x}^{k_b} p_x^{k_b} (1-p_x)^{N_x-k_b} = P(X=0) + P(X=1) + \cdots + P(X=N_x), \tag{4}$$

$$((1-p_x)-p_x)^{N_x} = \sum_{k_b=0}^{N_x} C_{N_x}^{k_b} (-p_x)^{k_b} (1-p_x)^{N_x-k_b} = P(X=0) - P(X=1) + \cdots + (-1)^{N_x} P(X=N_x). \tag{5}$$

Combining the two equations then gives

$$P(X \text{ is even}) = \frac{1}{2}\left(1 + (1-2p_x)^{N_x}\right), \tag{6}$$

$$P(X \text{ is odd}) = \frac{1}{2}\left(1 - (1-2p_x)^{N_x}\right). \tag{7}$$

□

1.2 Estimates of properties of memory traces for randomly drawn memory items

Estimates of firing probability of memory neurons after learning M randomly drawn memory items. To estimate the HD of memory traces for different input patterns, we need to estimate the probability that a memory neuron fires differently for different input patterns. To obtain this, we start with deriving a basic estimate called the firing probability, which represents the probability that a memory neuron fires in response to the memory item $\boldsymbol{x}$ after

Table 1. Definitions and explanations of key parameters used in the theoretical analysis.

Variables	Category	Explanations	Definitions
$f(*)$	General notations	Probability mass function of the binomial distribution	
$F(*)$		Cumulative distribution function of the binomial distribution	
w_{ji}		The synaptic weight between memory neuron j and input neuron i, which can be in an active state (1) or inactive state (0).	
$Q(\boldsymbol{x})$		Set of indices for memory neurons that receive a plateau potential for a given pattern $\boldsymbol{x}$ during learning	
$supp(\boldsymbol{x})$		Set which collects positions of 1's in $\boldsymbol{x}$	
$\iota(\boldsymbol{x})$		Size of $supp(\boldsymbol{x})$	
$S(j)$		Set which collects the index of input patterns (presented sequentially) for which the memory neuron j receives plateau potentials during learning	
$\boldsymbol{y}(j)$		The unit vector with only j-th bit position being 1	
c		Number of common 1's among non-orthogonal memory items	
f_q		Probability of plateau potential	
f_p		Input density	
f_w		Probability of connections	
f_d		Fraction of 1's being masked to zeros	
p_{f1}(or p_{f2})	Estimates for randomly drawn memory items	Conditional probability that a memory neuron fires in the recall phase when exposed to $\boldsymbol{x}$, given that the memory neuron receives (or does not receive) plateau for $\boldsymbol{x}$ during learning	Eqs.14 and 15
p_e(or p_o)		Conditional probability that a synapse connecting 1's of $\boldsymbol{x}$ and a memory neuron remains in an active state during the recall phase, given that the memory neuron receives (or does not receive) a plateau potential for $\boldsymbol{x}$	Eqs.12 and 13
$p_{f_{\text{avg}}}$		Probability that a memory neuron fires in the recall phase when exposed to $\boldsymbol{x}$	Eq.16
$\hat{f}_p$		Input density of the remaining bits in non-orthogonal memory items, excluding the common 1's	
c_w	Estimates for non-orthogonal memory items	The number of active synapses connecting common 1's to a memory neuron in the recall phase	
$\hat{p}_e$(or $\hat{p}_o$)		Conditional probability that a synapse connecting 1's of $\boldsymbol{x}$ and a memory neuron keeps in an active state in the recall phase, given that the memory neuron receives (or does not receive) plateau for $\boldsymbol{x}$ during learning	Eqs.22 and 23
$\hat{p}_{f1}$(or $\hat{p}_{f2}$)		Conditional probability of a memory neuron firing in response to $\boldsymbol{x}$ in the recall phase, given that the memory neuron receives (or does not receive) plateau for $\boldsymbol{x}$ during learning	Eqs.24 and 25
$\hat{p}_{f_{\text{avg}}}$		Probability that a memory neuron fires in the recall phase when exposed to $\boldsymbol{x}$	Eq.26
$\hat{p}_{c1}$(or $\hat{p}_{c2}$)		Conditional probability that memory neurons exhibit distinct firing patterns for $\boldsymbol{x}$ and $\boldsymbol{x}'$ in the recall phase, given that the memory neuron receives (or does not) plateau for $\boldsymbol{x}$ during learning	Eqs.27 and 28
$\hat{s}_j$		The number of weight updates of a synapse connecting common 1's to the memory neuron j throughout learning phase	

learning M patterns. Note that the firing behaviour of a memory neuron is determined by whether the weighted sum of binary inputs is greater than a firing threshold v_{th}. It indicates that the firing probability depends on the number of weights that are in the active status (i.e., $w_{ji} = 1$) after the learning is completed. Therefore, to estimate this firing probability, we need to model the effect of BTSP on the active state of w_{ji} after learning.

We introduce the unit vector $\boldsymbol{y}(j) \in \mathbb{R}^{n\times 1}$ with only the j bit position as 1 for representing the binary plateau pattern. For simplicity, we assume that the initial weights are set to a zero matrix and use the term $\boldsymbol{W}$ to refer to the weights after learning M patterns. The weight matrix $\boldsymbol{W}$ can be written as the summation of multiple weight updates, organized in the order of the bit position j, as follows:

$$\boldsymbol{W} = \sum_{j=1}^{n} \boldsymbol{y}(j) \times \hat{1}(\sum_{k\in S(j)} \boldsymbol{x}_k^T), \tag{8}$$

where the set $S(j)$ collects the index of patterns for which the memory neuron j receives the plateau potentials during the learning process. $\hat{1}(x)$ is an indicator function which models the impact of the parity of weight update:

$$\hat{1}(x) = \begin{cases} 1, & \text{if } x \text{ is odd,} \\ 0, & \text{if } x \text{ is even.} \end{cases} \tag{9}$$

Next, we calculate the weighted sum of inputs $\boldsymbol{u}(\boldsymbol{x})$:

$$\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{W} \times \boldsymbol{x} = \sum_{j=1}^{n} \left(\boldsymbol{y}(j) \times \hat{1}(\sum_{k\in S(j)} \boldsymbol{x}_k^T) \right) \times \boldsymbol{x} = \sum_{j=1}^{n} \boldsymbol{y}(j) \times \left(\hat{1}(\sum_{k\in S(j)} \boldsymbol{x}_k^T) \times \boldsymbol{x} \right). \tag{10}$$

where $\boldsymbol{x}_k$ represents the memory item for which the memory neuron j receives a plateau potential during the learning phase.

Let u_j be the j_{th} element of $\boldsymbol{u}$. According to the definition of $\boldsymbol{y}(j)$, we have

$$u_j(\boldsymbol{x}) = \hat{1}(\sum_{k\in S(j)} \boldsymbol{x}_k^T) \times \boldsymbol{x} = \sum_{i\in supp(\boldsymbol{x})} \hat{1}(\sum_{k\in S(j)} x_{ki}), \tag{11}$$

where x_{ki} is the element i in $\boldsymbol{x}_k$.

Note that $\hat{1}(\sum_{k\in S(j)} x_{ki})$ is a binary variable that represents the active state of w_{ji}. According to Eq.1, its value is determined by the parity of actual weight updates during the learning phase. Due to the assumption of the input density f_p, plateau probability f_q, and the weight update probability (0.5), the probability of weight update can be modeled by Bernoulli distribution with a probability of $f_p f_q/2$. Therefore, we can estimate the probability $P(\hat{1}(\sum_{k\in S(j)} x_{ki}) = 1)$ in the following two cases:

- For any $j_1 \in Q(\boldsymbol{x})$ and any $i \in supp(\boldsymbol{x})$, we estimate the probability p_e that $w_{j_1 i}$ is active after learning M patterns:

$$p_e := P\big(\hat{1}(\sum_{k\in S(j_1)} x_{ki}) = 1|\boldsymbol{x}\big) = P\big(\sum_{k\in S(j_1)} x_{ki} \text{ is odd}|\boldsymbol{x}\big) = P\big(\sum_{k\in S(j_1)} x_{ki} - 1 \text{ is even}\big) = \frac{1}{2}\big(1 + (1 - f_p f_q)^{M-1}\big), \tag{12}$$

  where the estimate of the parity in the last equality uses Lemma 1.

- For any $j_2 \notin Q(\boldsymbol{x})$ and any $i \in supp(\boldsymbol{x})$, we estimate the probability p_o that $w_{j_2 i}$ is active after learning M patterns:

$$p_o := P\big(\hat{1}(\sum_{k \in S(j_2)} x_{ki}) = 1 | \boldsymbol{x}\big) = P(\sum_{k \in S(j_2)} x_{ki} \text{ is odd} | \boldsymbol{x}) = \frac{1}{2}\big(1 - (1 - f_p f_q)^{M-1}\big). \tag{13}$$

The above equations provide estimates of the probability that any given connection is in an active state after learning M patterns. In order to incorporate the effects of the inherited connection probability f_w, we directly multiply f_w by p_o (or p_e) since they are independent. We then employ the cumulative binomial probability F to estimate the firing probability of memory neurons given the size of support set $\iota(\boldsymbol{x})$ as I:

- For any $j_1 \in Q(\boldsymbol{x})$, we have:

$$p_{f1}(j_1, \boldsymbol{x}, I) := P\left(u_{j_1} > v_{th} | \boldsymbol{x}, I, j_1 \in Q(\boldsymbol{x})\right) = 1 - F(v_{th}, I, f_w p_e). \tag{14}$$

- For any $j_2 \notin Q(\boldsymbol{x})$, we have:

$$p_{f2}(j_2, \boldsymbol{x}, I) := P\left(u_{j_2} > v_{th} | \boldsymbol{x}, I, j_2 \notin Q(\boldsymbol{x})\right) = 1 - F(v_{th}, I, f_w p_o). \tag{15}$$

The above Eqs.14 and 15 show that the formulas of firing probability do not depend on the specific positions of j_1 and j_2. For brevity, we will omit the indices j_1 and j_2 of the brackets in the following derivations.

Given the estimates of p_{f1} and p_{f2}, we can finally obtain the firing probability of a memory neuron by applying the total probability rule, which allows us to relax the condition of the given size $\iota(\boldsymbol{x})$ of the support set:

$$\begin{aligned}
p_{f_{avg}}(\boldsymbol{x}) &= P(u_j > v_{th} | \boldsymbol{x}) \\
&= \sum_{I=0}^{m} P(u_j > v_{th}, \iota(\boldsymbol{x}) = I | \boldsymbol{x}) \\
&= \sum_{I=0}^{m} P(\iota(\boldsymbol{x}) = I) P(u_j > v_{th} | \iota(\boldsymbol{x}) = I, \boldsymbol{x}) \\
&= \sum_{I=0}^{m} P(\iota(\boldsymbol{x}) = I) \Big(P(j \in Q(\boldsymbol{x})) P(u_j > v_{th} | j \in Q(\boldsymbol{x})) + P(j \notin Q(\boldsymbol{x})) P(u_j > v_{th} | j \notin Q(\boldsymbol{x})) \Big) \\
&= \sum_{I=0}^{m} C_m^I f_p^I (1 - f_p)^{m-I} \Big(f_q p_{f1}(\boldsymbol{x}, I) + (1 - f_q) p_{f2}(\boldsymbol{x}, I) \Big).
\end{aligned} \tag{16}$$

Estimates of expected HD between memory traces for the original and masked memory items. In this section, we utilize the properties of p_e and p_o to estimate the expected HD between memory traces for the original and masked memory items.

Suppose $\boldsymbol{x}$ is a random binary memory item with the input density of f_d and $\boldsymbol{x'}$ is the masked pattern of $\boldsymbol{x}$ with the fraction f_d of 1's being masked to zeros. We formulate the expected HD in terms of the probability that individual

neurons fire differently for any complete memory item and its partially masked pattern:

$$
\begin{aligned}
&E(HD(\boldsymbol{z}(\boldsymbol{x}),\boldsymbol{z}(\boldsymbol{x}')))\\
&=\sum_{j=1}^{n} P(z_j(\boldsymbol{x})\neq z_j(\boldsymbol{x}'))\\
&=\sum_{j=1}^{n} P(j\in Q(\boldsymbol{x}))P\big(z_j(\boldsymbol{x})\neq z_j(\boldsymbol{x}')|j\in Q(\boldsymbol{x})\big)+P(j\notin Q(\boldsymbol{x}))P\big(z_j(\boldsymbol{x})\neq z_j(\boldsymbol{x}')|j\notin Q(\boldsymbol{x})\big)\\
&=nf_qP\big(z_j(\boldsymbol{x})\neq z_j(\boldsymbol{x}')|j\in Q(\boldsymbol{x})\big)+n(1-f_q)P\big(z_j(\boldsymbol{x})\neq z_j(\boldsymbol{x}')|j\notin Q(\boldsymbol{x})\big)
\end{aligned}
\tag{17}
$$

where $Q(\boldsymbol{x})$ refers to the set of indices of the memory neuron that got plateau potentials for $\boldsymbol{x}$ during the learning process. Eq.17 indicates that to estimate this expression $E\big(HD(\boldsymbol{z}(\boldsymbol{x}),\boldsymbol{z}(\boldsymbol{x}'))\big)$, we only need to estimate the probability that a memory neuron fires differently for $\boldsymbol{x}$ and $\boldsymbol{x}'$ for both the set $Q(\boldsymbol{x})$ and its complement.

By using p_{f_1} and p_{f_2} as defined by Eqs.14 and 15, we estimate $P(z_j(\boldsymbol{x})\neq z_j(\boldsymbol{x}'))$ in the following two separate cases:

- For any $j_1\in Q(\boldsymbol{x})$, we have:

$$
\begin{aligned}
&P\big(z_{j_1}(\boldsymbol{x})\neq z_{j_1}(\boldsymbol{x}')|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\\
&=P\big(z_{j_1}(\boldsymbol{x})=1,z_{j_1}(\boldsymbol{x}')=0|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)+P\big(z_{j_1}(\boldsymbol{x})=0,z_{j_1}(\boldsymbol{x}')=1|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\\
&=P\big(z_{j_1}(\boldsymbol{x})=1,z_{j_1}(\boldsymbol{x}')=0|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\quad(\text{Note that } P(z_{j_1}(\boldsymbol{x})=0,z_{j_1}(\boldsymbol{x}')=1)=0)\\
&\approx P\big(z_{j_1}(\boldsymbol{x})=1|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)P\big(z_{j_1}(\boldsymbol{x}')=0|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\\
&=\big(1-F(v_{th},I,f_wp_e)\big)F\big(v_{th},I(1-f_d),f_wp_e\big).
\end{aligned}
\tag{18}
$$

  The above estimate approximates $P(z_{j_1}(\boldsymbol{x})=1,z_{j_1}(\boldsymbol{x}')=0)$ as two independent terms, $P(z_{j_1}(\boldsymbol{x})=1)$ and $P(z_{j_1}(\boldsymbol{x}')=1)$. It ignores the dependence on the same parity of the weight updates indicated by Eqs.14 and 15. Nevertheless, our numerical estimates show good consistency using the simple strategy. A more precise but complicated estimate is provided in the following derivation for the non-orthogonal memory items (Eqs.27 and 28), where we further employ the total probability rule to deal with this dependency and make this estimate applicable for the common 1's.

- For any $j_2\notin Q(\boldsymbol{x})$, similarly, we have

$$
\begin{aligned}
&P\big(z_{j_2}(\boldsymbol{x})\neq z_{j_2}(\boldsymbol{x}')|j_2\notin Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\\
&\qquad=P\big(z_{j_2}(\boldsymbol{x})=1,z_{j_2}(\boldsymbol{x}')=0|j_2\notin Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\\
&\qquad\approx F(v_{th},I(1-f_d),f_wp_o)\big(1-F(v_{th},I,f_wp_o)\big).
\end{aligned}
\tag{19}
$$

By combining Eqs.18 and 19, we can obtain estimates for the expected HD,

$$
\begin{aligned}
&E(HD(\boldsymbol{z}(\boldsymbol{x}),\boldsymbol{z}(\boldsymbol{x}')))\\
&=\sum_{I=0}^{m} Pr(\iota(\boldsymbol{x})=I)E\big(HD(\boldsymbol{z}(\boldsymbol{x}),\boldsymbol{z}(\boldsymbol{x}'))|\iota(\boldsymbol{x})=I\big)\\
&=\sum_{I=0}^{m} nPr(\iota(\boldsymbol{x})=I)\Big(f_qP\big(z_{j_1}(\boldsymbol{x})\neq z_{j_1}(\boldsymbol{x}')|j_1\in Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)+(1-f_q)P\big((z_{j_2}(\boldsymbol{x}))\neq z_{j_2}(\boldsymbol{x}')|j_2\notin Q(\boldsymbol{x}),\iota(\boldsymbol{x})=I\big)\Big)\\
&\approx n\sum_{I=0}^{m} C_m^I f_p^I(1-f_p)^{m-I}\Big(f_qF(v_{th},I(1-f_d),f_wp_e)\big(1-F(v_{th},I,f_wp_e)\big)+(1-f_q)F(v_{th},I(1-f_d),f_wp_o)\big(1-F(v_{th},I,f_wp_o)\big)\Big).
\end{aligned}
\tag{20}
$$

$E\big(HD(\boldsymbol{z}(\boldsymbol{x}),\boldsymbol{z}(\boldsymbol{x}'))\big)$ provides an estimate of dissimilarity between the memory traces for the original and masked memory items, which was used in Fig. 2C.

Estimates of expected HD between memory traces for different randomly drawn memory items. To estimate the average HD between memory traces of two different memory items, we use $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ to represent two different orthogonal patterns. Owing to the assumption of randomly drawn memory items, we can directly utilize $p_{f_{avg}}$ to estimate $HD(\boldsymbol{z}(\boldsymbol{x}_a), \boldsymbol{z}(\boldsymbol{x}_b))$ as follows,

$$
\begin{aligned}
& E\big(HD(\boldsymbol{z}(\boldsymbol{x}_a), \boldsymbol{z}(\boldsymbol{x}_b))\big) \\
&= \sum_{j=1}^{n} P\big(z_j(\boldsymbol{x}_a) \neq z_j(\boldsymbol{x}_b)\big) \\
&= \sum_{j=1}^{n} P\big(z_j(\boldsymbol{x}_a) = 1, z_j(\boldsymbol{x}_b) = 0\big) + P\big(z_j(\boldsymbol{x}_a) = 0, z_j(\boldsymbol{x}_b) = 1\big) \\
&= \sum_{j=1}^{n} P(z_j(\boldsymbol{x}_a) = 1) P(z_j(\boldsymbol{x}_b) = 0) + P(z_j(\boldsymbol{x}_a) = 0) P(z_j(\boldsymbol{x}_b) = 1) \\
&= 2n p_{f_{avg}} (1 - p_{f_{avg}}).
\end{aligned}
\tag{21}
$$

Eq.21 provides the estimates of the memory traces for different original memory items. By combining the estimates in Eq.20 and Eq.21, we can estimate the ratio of HD used in Fig. 2C.

1.3 Estimates of properties of memory traces for non-orthogonal memory items

Estimates of firing probability of memory neurons after learning M non-orthogonal memory items. For non-orthogonal memory items, the approach to estimating firing probabilities is similar to that for randomly drawn memory items, but with an additional consideration for the impact of BTSP on common 1's shared among memory items.

Consider a random binary memory item $\boldsymbol{x}$ with c common 1's shared across non-orthogonal memory items, and the remaining elements having an input density of $\hat{f}_p$. Unlike the case with randomly drawn memory items, here the probability of synaptic weight updates associated with these common 1's needs to be recalculated. The active state of each synaptic weight w_{ji} is still determined by the parity of updates; however, the probability of these updates is altered due to the repeated occurrence of common 1's. We now derive the conditional probability $P\big(\hat{1}(\sum_{k \in S(j)} x_{ki}) = 1 | S_j\big)$ given the size s_j of $S(j)$ in two cases:

- For any $j_1 \in Q(\boldsymbol{x})$ and any $i \in supp(\boldsymbol{x})$, we estimate the probability $\hat{p}_e(s_{j_1})$ that $w_{j_1 i}$ is active given the size s_{j_1} of $S(j_1)$:

$$
\hat{p}_e(s_{j_1}) := P\big(\hat{1}(\sum_{k \in S(j_1)} x_{ki}) = 1 | \boldsymbol{x}, s_{j_1}\big) = P(\sum_{k \in S(j_1)} x_{ki} - 1 \text{ is even} | s_{j_1}) = \frac{1}{2}\big(1 + (1 - \hat{f}_p)^{s_{j_1} - 1}\big). \tag{22}
$$

- For any $j_2 \notin Q(\boldsymbol{x})$ and any $i \in supp(\boldsymbol{x})$, we estimate the probability $\hat{p}_o(s_{j_2})$ that $w_{j_2 i}$ is active given the size s_{j_2} of $S(j_2)$:

$$
\hat{p}_o(s_{j_2}) := P\big(\hat{1}(\sum_{k \in S(j_2)} x_{ki}) = 1 | \boldsymbol{x}, s_{j_2}\big) = P(\sum_{k \in S(j_2)} x_{ki} \text{ is odd} | s_{j_2}) = \frac{1}{2}\big(1 - (1 - \hat{f}_p)^{s_{j_2}}\big). \tag{23}
$$

With the estimates of the probability for synaptic weights remaining active after learning, we now proceed to estimate the firing probability. In order to model the effect of common bits, we introduce the following two variables: (1) c_w, denoting the number of active connections from input neurons receiving common 1's to a given memory neuron

after learning, and (2) $\hat{s}_j$, representing the number of weight updates for a synapse connecting neuron receiving common 1's during the learning. Recall the dependence of the active state of the weights on the parity of the number of weight updates: if $\hat{s}_j$ is odd, then the synapse will be in the active state after the learning and thereby the partial weighted sum of inputs from the common 1's is equal to c_w; otherwise, the synapse will be in the inactive state, contributing zero to the weighted sum. By doing so, the introduction of c_w and $\hat{s}_j$ can allow for a nuanced analysis of the firing probabilities, factoring in the distinct impact of common 1's on synaptic behavior. We now evaluate the conditional firing probability of memory neurons to a given memory item post-learning as follows,

- For any $j_1 \in Q(\boldsymbol{x})$, we have

$$
\begin{aligned}
&\hat{p}_{f1}(\boldsymbol{x}, I, c_w, c) \\
&= P\big(z_{j_1}(\boldsymbol{x}) = 1 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \\
&= \sum_{l=0}^{M} P\big(z_{j_1}(\boldsymbol{x}) = 1, s_{j_1} = l | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \quad \text{(by the total probability rule)} \\
&= \sum_{l=0}^{M} P(s_{j_1} = l) P\big(z_{j_1}(\boldsymbol{x}) = 1 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l\big) \quad \text{(by considering the parity of } \hat{s}_{j_1}) \\
&= \sum_{l=0}^{M} P(s_{j_1} = l) \Big(P\big(z_{j_1}(\boldsymbol{x}) = 1, \hat{s}_{j_1} \text{is even} | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l\big) + P\big(z_{j_1}(\boldsymbol{x}) = 1, \hat{s}_{j_1} \text{is odd} | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l\big) \Big) \\
&= \sum_{l=0}^{M} P(s_{j_1} = l) P(\hat{s}_{j_1} \text{is odd}) P\big(z_{j_1}(\boldsymbol{x}) = 1 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l, \hat{s}_{j_1} \text{is odd}\big) \\
&+ \sum_{l=0}^{M} P(s_{j_1} = l) P(\hat{s}_{j_1} \text{is even}) P\big(z_{j_1}(\boldsymbol{x}) = 1 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l, \hat{s}_{j_1} \text{is even}\big) \\
&= \sum_{l=0}^{M} P(s_{j_1} = l) \Big(\frac{1}{2}\big(1 - F(v_{th} - c_w, I - c, f_w \hat{p}_e(l))\big) + \frac{1}{2}\big(1 - F(v_{th}, I - c, f_w \hat{p}_e(l))\big) \Big) \\
&= \sum_{l=0}^{M} C_M^l f_q^l (1 - f_q)^{M-l} \Big(\frac{1}{2}\big(1 - F(v_{th} - c_w, I - c, f_w \hat{p}_e(l))\big) + \frac{1}{2}\big(1 - F(v_{th}, I - c, f_w \hat{p}_e(l))\big) \Big) \\
&\approx \sum_{l=0}^{M} \frac{(M f_q)^l}{2 e^{M f_q} l!} \Big(2 - F\big(v_{th} - c_w, I - c, f_w \hat{p}_e(l)\big) - F\big(v_{th}, I - c, f_w \hat{p}_e(l)\big) \Big).
\end{aligned}
\tag{24}
$$

The last equation uses the Poisson approximation to the binomial probability for simplifying analysis.

- For any $j_2 \notin Q(\boldsymbol{x})$, we can obtain the estimate in a similar manner

$$
\begin{aligned}
&\hat{p}_{f2}(\boldsymbol{x}, I, c_w, c) \\
&= P(z_{j_2}(\boldsymbol{x}) = 1 | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I) \\
&= \sum_{l=0}^{M} P\big(z_{j_2}(\boldsymbol{x}) = 1, s_{j_2} = l, j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \\
&= \sum_{l=0}^{M} P(s_{j_2} = l) P\big(z_{j_2}(\boldsymbol{x}) = 1 | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_2} = l\big) \quad \text{(by considering the parity of } \hat{s}_{j_2}) \\
&= \sum_{l=0}^{M} P(s_{j_2} = l) \Big(P\big(z_{j_2}(\boldsymbol{x}) = 1, \hat{s}_{j_2} \text{is even} | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_2} = l\big) + P\big(z_{j_2}(\boldsymbol{x}) = 1, \hat{s}_{j_2} \text{is odd} | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_2} = l\big) \Big) \\
&= \sum_{l=0}^{M} C_M^l f_q^l (1 - f_q)^{M-l} \Big(\frac{1}{2}\big(1 - F(v_{th}, I - c, f_w \hat{p}_o(l))\big) + \frac{1}{2}\big(1 - F(v_{th} - c_w, I - c, f_w \hat{p}_o(l))\big) \Big) \\
&\approx \sum_{l=0}^{M} \frac{(M f_q)^l}{2 e^{M f_q} l!} \Big(2 - F\big(v_{th}, I - c, f_w \hat{p}_o(l)\big) - F\big(v_{th} - c_w, I - c, f_w \hat{p}_o(l)\big) \Big).
\end{aligned}
$$

(25)

These estimations account for the firing probabilities of memory neurons post-learning. Combining Eqs.24 and 25 gives the estimate of the average firing probability of memory neurons

$$
\begin{aligned}
&\hat{p}_{f_{avg}}(\boldsymbol{x},c) \\
&= \sum_{I'=0}^{m-c}\sum_{c_w=0}^{c} P\big(\imath(\boldsymbol{x}) = I'+c, c_w\big) P\big(u_j > v_{th} | \imath(\boldsymbol{x}) = I'+c, c_w\big) \\
&= \sum_{I'=0}^{m-c}\sum_{c_w=0}^{c} P(\imath(\boldsymbol{x}) = I'+c) P(c_w) \Big(f_q \hat{p}_{f1}\big(\boldsymbol{x}, I'+c, c_w, c\big) + (1-f_q)\hat{p}_{f2}(\boldsymbol{x}, I'+c, c_w, c)\Big) \\
&= \sum_{I'=0}^{m-c}\sum_{c_w=0}^{c} C_{m-c}^{I'} C_c^{c_w} \hat{f}_p^{I'} (1-\hat{f}_p)^{m-c-I'} f_w^{c_w} (1-f_w)^{c-c_w} \big(f_q \hat{p}_{f1}(\boldsymbol{x}, I'+c, c_w, c) + (1-f_q)\hat{p}_{f2}(\boldsymbol{x}, I'+c, c_w, c)\big),
\end{aligned}
\tag{26}
$$

Estimates of expected HD between memory traces for the original and masked memory items. We use the firing probability of memory neurons to estimate the expected HD between memory traces for the original and masked memory items. As indicated by Eq.18, to estimate $E\big(HD(\boldsymbol{z}(\boldsymbol{x}), \boldsymbol{z}(\boldsymbol{x'}))\big)$, we only need to estimate $P(z_j(\boldsymbol{x}) = 1, z_j(\boldsymbol{x'}) = 0 | \imath(\boldsymbol{x}) = I)$.

Unlike the estimate of randomly drawn memory items, the interaction of common bits and masking bits complicates the estimates. For simplification, we use the average number of masking common 1's, cf_d, to approximate the effects of the masking fraction on common 1's. On this basis, we evaluate $P(z_j(\boldsymbol{x}) = 1, z_j(\boldsymbol{x'}) = 0 | \imath(\boldsymbol{x}) = I)$ in two different cases, $j_1 \in Q(\boldsymbol{x})$ and $j_2 \notin Q(\boldsymbol{x})$.

- For any $j_1 \in Q(\boldsymbol{x})$, we follow the derivation of Eq.24 and have

$$
\begin{aligned}
&\hat{p}_{c1}(j_1, \boldsymbol{x}, \boldsymbol{x'}, I, c_w) \\
&= P\big(z_{j_1}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \\
&= \sum_{l=0}^{M} P\big(z_{j_1}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0, s_{j_1} = l | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \\
&= \sum_{l=0}^{M} P(s_{j_1} = l) P\big(z_{j_1}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l\big) \\
&= \sum_{l=0}^{M} P(s_{j_1} = l) P(\hat{s}_{j_1} \text{is even}) P\big(z_{j_1}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l, \hat{s}_{j_1} \text{is even}\big) \\
&+ \sum_{l=0}^{M} P(s_{j_1} = l) P(\hat{s}_{j_1} \text{is odd}) P\big(z_{j_1}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0 | j_1 \in Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_1} = l, \hat{s}_{j_1} \text{is odd}\big) \\
&\approx \sum_{l=0}^{M} \frac{(Mf_q)^l}{2e^{Mf_q} l!} \Big(\big(1 - F(v_{th} - c_w, I-c, f_w \hat{p}_e(l))\big) F\big(v_{th} - c_w(1-f_d), (I-c)(1-f_d), f_w \hat{p}_e(l)\big) \\
&+ \big(1 - F(v_{th}, I-c, f_w \hat{p}_e(l))\big) F\big(v_{th}, (I-c)(1-f_d), f_w \hat{p}_e(l)\big)\Big),
\end{aligned}
\tag{27}
$$

where the handling of parity of s_{j_1} follows the same as Eq.24 and thus we simplify the derivation here.

- For any $j_2 \notin Q(\boldsymbol{x})$, we can derive the estimate in a similar manner:

$$
\begin{aligned}
&\hat{p}_{c2}(j_2, \boldsymbol{x}, \boldsymbol{x'}, I, c_w) \\
&= P\big(z_{j_2}(\boldsymbol{x}) = 1, z_{j_2}(\boldsymbol{x'}) = 0 | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \\
&= \sum_{l=0}^{M} P\big(z_{j_2}(\boldsymbol{x}) = 1, z_{j_2}(\boldsymbol{x'}) = 0, s_{j_2} = l | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I\big) \\
&= \sum_{l=0}^{M} P(s_{j_2} = l) P\big(z_{j_2}(\boldsymbol{x}) = 1, z_{j_2}(\boldsymbol{x'}) = 0 | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_2} = l\big) \\
&= \sum_{l=0}^{M} P(s_{j_2} = l) P(\hat{s}_{j_2} \text{is even}) P\big(z_{j_2}(\boldsymbol{x}) = 1, z_{j_2}(\boldsymbol{x'}) = 0 | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_2} = l, \hat{s}_{j_2} \text{is even}\big) \\
&+ \sum_{l=0}^{M} P(s_{j_2} = l) P(\hat{s}_{j_2} \text{is odd}) P\big(z_{j_2}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0 | j_2 \notin Q(\boldsymbol{x}), \imath(\boldsymbol{x}) = I, s_{j_2} = l, \hat{s}_{j_2} \text{is odd}\big) \\
&\approx \sum_{l=0}^{M} \frac{(Mf_q)^l}{2e^{Mf_q} l!} \Big(\big(1 - F(v_{th}, I - c, f_w \hat{p}_o(l))\big) F\big(v_{th}, (I-c)(1-f_d), f_w \hat{p}_o(l)\big) \\
&+ \big(1 - F(v_{th} - c_w, I - c, f_w \hat{p}_o(l))\big) F\big(v_{th} - c_w(1-f_d), (I-c)(1-f_d), f_w \hat{p}_o(l)\big) \Big).
\end{aligned}
\tag{28}
$$

As the above Eqs.27 and 28 do not depend on the specific positions of j_1 and j_2, we omit the indices j_1 and j_2 in the following derivations.

Combining Eqs.27 and 28 gives the estimate of $E\big(HD(\boldsymbol{z}(\boldsymbol{x}), \boldsymbol{z}(\boldsymbol{x'}))\big)$

$$
\begin{aligned}
&E\big(HD(\boldsymbol{z}(\boldsymbol{x}), \boldsymbol{z}(\boldsymbol{x'}))\big) \\
&= n \sum_{I=c}^{m} Pr(\imath(\boldsymbol{x}) = I) P\big(z_j(\boldsymbol{x}) = 1, z_j(\boldsymbol{x'}) = 0, \forall j | \imath(\boldsymbol{x}) = I\big) \\
&= n \sum_{I=c}^{m} Pr(\imath(\boldsymbol{x}) = I) \Big(f_q P\big(z_{j_1}(\boldsymbol{x}) = 1, z_{j_1}(\boldsymbol{x'}) = 0 | \imath(\boldsymbol{x}) = I, j_1 \in Q(\boldsymbol{x})\big) \\
&+ (1 - f_q) \big(P(z_{j_2}(\boldsymbol{x}) = 1, z_{j_2}(\boldsymbol{x'}) = 0 | \imath(\boldsymbol{x}) = I, j_2 \notin Q(\boldsymbol{x}))\big) \Big) \\
&\approx n \sum_{I'=0}^{m-c} \sum_{c_w=0}^{c} C_{m-c}^{I'} C_c^{c_w} \hat{f}_p^{I'} (1 - \hat{f}_p)^{m-c-I'} f_w^{c_w} (1 - f_w)^{c-c_w} \Big(f_q \hat{p}_{c1}\big(\boldsymbol{x}, \boldsymbol{x'}, I' + c, c_w\big) + (1 - f_q) \hat{p}_{c2}\big(\boldsymbol{x}, \boldsymbol{x'}, I' + c, c_w\big) \Big).
\end{aligned}
\tag{29}
$$

$E\big(HD(\boldsymbol{z}(\boldsymbol{x}), \boldsymbol{z}(\boldsymbol{x'}))\big)$ provides the estimate of dissimilarity between the memory traces for the original and masked memory items, which was used in Fig. 4B.

Estimates of expected HD between memory traces for different non-orthogonal memory items. To estimate the expected HD of memory traces between non-orthogonal memory items, we use $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ to represent two different non-orthogonal memory items. By using the estimate of firing probability for non-orthogonal memory

items, we can directly obtain an approximate estimate for this HD:

$$
\begin{aligned}
&E\big(HD(\boldsymbol{z}(\boldsymbol{x}_a),\boldsymbol{z}(\boldsymbol{x}_b))\big)\\
&=\sum_{j=1}^{n} P\big(z_j(\boldsymbol{x}_a)=1,z_j(\boldsymbol{x}_b)=0\big)+P\big(z_j(\boldsymbol{x}_a)=0,z_j(\boldsymbol{x}_b)=1\big)\\
&\approx\sum_{j=1}^{n} P(z_j(\boldsymbol{x}_a)=1)P(z_j(\boldsymbol{x}_b)=0)+P(z_j(\boldsymbol{x}_a)=0)P(z_j(\boldsymbol{x}_b)=1)\\
&=2n\hat{p}_{f_{avg}}(\boldsymbol{x}_a,c)\big(1-\hat{p}_{f_{avg}}(\boldsymbol{x}_a,c)\big).
\end{aligned}
\tag{30}
$$

In the above estimation, we disregard the correlation arising from common bits for simplicity. This approximation is suitable for cases where the fraction of common 1's is small and the equal sign can be taken when the fraction of common 1's is 0. A more precise but intricate estimate can be obtained by expanding the estimation along $S(j)$, as exemplified by Eqs.24 and 25. The estimated HD in Eq.30 provides an approximation of the memory traces for different original memory items. By combining this estimate with the estimate in Eq.29, we can directly obtain the estimate of the ratio of HD used in Fig. 4B.

2 Details of experimental setup and simulations

Detailed justification of the stochastic aspect of the BTSP rule of Eq. 1. The probability 0.5 occurs in our BTSP rule because the experimental data show that if the arrival time of the input pattern $\boldsymbol{x}$ falls into the interior window from -3 to 2s relative to the onset of the plateau potential, then LTP is applied to all synaptic weights whose value is not yet maximal. If the arrival time of the input pattern falls into the outer parts of the 10s window, i.e., into the window from -6 to -3s or from 2 to 4s relative to the onset of the plateau, then LTD is applied to all synapses whose weight is not yet minimal. Since there is no obvious dependence between the arrival times of the input pattern and the onsets of plateau potentials, we assume that the arrival time of the input pattern falls with probability 0.5 into the inner part of the 10s window, and with probability 0.5 into the outer part. Hence with probability 0.5 a synaptic weight is not updated when an input pattern arrives within the window from -6 to 4s relative to the onset of a plateau potential because the arrival time of the input pattern falls into the "wrong" part of this 10s window where no update of its current binary weight value is possible.

Details of simulation of continuous weight changes by BTSP rule (Fig. 1D and E). In Fig. 1D, continuous weight changes were simulated using Eq. 1 from (Milstein et al., 2021), with parameters: $w^{max}=8$, $\tau_{ET}=1000ms$, $\tau_{IS}=500ms$, $k^+=0.5$, $k^-=0.5$, $\alpha^+=0.01$, $\beta^+=35$, $\alpha^-=0.01$, $\beta^-=800$. For Fig. 1E, two neurons connected by eight synapses with binary synaptic weights were modeled. The initial binary weights were randomly sampled from a Bernoulli distribution(probability 0.5). The experiment involved 10,000 simulations, with results indicating average weight changes.

Details of analysis of the masking fraction for randomly drawn memory items (Fig. 2). Unless otherwise specified, in all our experiments, we initialized the feedforward weights with a zero matrix and employed the following default parameters for modelling: $m=25,000$, $n=39,000$, $f_p=0.005$, $f_q=0.005$, $f_w=0.6$. For the data points in the simulation, we sampled M at intervals of 2000 and sampled the masking fraction at intervals of 0.02. Theoretical predictions were made for the same data points as those used in the simulations.

For every masking fraction f_d, memory items $\boldsymbol{x}'$ were generated by randomly masking a fraction f_d of the 1's in $\boldsymbol{x}$. We selected the optimal v_{th} by minimizing the predicted HD ratio (i.e., relative dissimilarity). Throughout the optimization process, we used a fixed masking fraction of 33% and the maximum number of input patterns M (i.e.,

30,000) for the predicted HD ratio by Eqs. 16 and 26. The prediction for panel C was obtained using Eq. 16,the prediction for panel D using Eqs. 13, and the prediction for panel E using Eqs. 20 and 16.

Details of comparison with other memory models (Fig. 3). We built the BTSP, HFN, and RP models with the same input size and connection probability. To establish the HFN with sparse binary inputs, we followed the previous studies (see Eq.5 of (Tsodyks and Feigel'man, 1988)) and formalized the weight matrix $\boldsymbol{W}^H$ of the HFN by

$$w_{ji}^H = \begin{cases} 0, & i = j, \\ \frac{1}{m}\sum_{h=1}^{M}(x_{hi} - f_p)(x_{hj} - f_p), & i \neq j. \end{cases} \tag{31}$$

During the recall phase, we searched for the optimal threshold of the state update function (see Eq.3 of (Tsodyks and Feigel'man, 1988)) from $[0.0, 0.02]$ with the minimum step 0.001. Referring to the memory capacity defined in section 17.2.4 of (Gerstner et al., 2014), we adopted an optimization criterion that maximizes the number of patterns that HFN can recall correctly within the error tolerance $m * 0.001$ for the dissimilarity (i.e., HD) between the memory traces of the original pattern and the masked pattern. We used the resulting optimal threshold 0.001 for retrieval. We used the synchronous update rule and iterated the state of all neurons 100 times for retrieval. We employed a fixed threshold to binarize the HFN's weights and considered three binarization threshold values, including (1) zero, (2) the mean of $\boldsymbol{W}^H$, and (3) the median of $\boldsymbol{W}^H$. For each binarization threshold, we applied the grid search for optimizing the threshold of state function for the binarized HFN. The resulting optimal threshold, $v_{th}^H = 0.001$, was adopted in Panel G. Further details on the binarization methods and search results are provided in the Supplementary Information and Fig.S7.

We constructed the RP using binary connection weights. The weights were initialized with the same fraction of weights with value 1 created by BTSP after learning 30,000 patterns (see Fig. 2D for the fraction). We binarized the projection results by applying a fixed threshold which was optimized using a grid search approach by simulation, employing the same optimization criterion as for the BTSP.

Details of analysis of the masking fraction for non-orthogonal memory items (Fig. 4). For each input pattern, we generated common 1's corresponding to the number $c = \lfloor m f_c f_p \rfloor$. The remaining bits were then generated at a density of $\hat{f}_p = (m f_p - c)/(m - c)$, ensuring the average number of 1's per pattern matched the value $m f_p$ used for randomly drawn memory items. Corresponding data sets were generated for each specific value of c, and 33% of elements in each non-orthogonal memory item were randomly masked during the recall phase. Predicted ratios were derived from Eqs.29 and 30. For the HFN, optimization of the neuron threshold was executed through a grid search in the range [0.0, 0.02], with a step of 0.001. Optimal performance, as shown in Panel C, was achieved with a threshold value of $v_{th} = 0.002$.

Details of experimental setup for the input completion (Fig. 5). For the sake of simplicity, we trained direct synaptic connections from the memory neurons to the input neurons with a simple Hebbian-like learning rule during the learning phase for the M memory items. Each input pattern $\boldsymbol{x}$ is presented once, as before, but we now let it persist for two steps. In the first step, random plateau signals are generated in the memory neurons and the feedforward connections from the input to the memory neurons are adjusted via BTSP. In the second step, the feedback connection weights were adjusted via a fast Hebbian-like learning mechanism: if both the pre-and postsynaptic neuron of a feedback connection assume value 1, the synaptic weight of this feedback connection is incremented by 1; otherwise it remains unchanged. If the weight value exceeded one, it was set to a saturation value of one. These feedback connections have a density of 0.6, as for the feedforward connection. To enhance the performance in practical applications, we employed a linear function ($v_f = \alpha x + \beta$), with the average fraction of 1's as the input variable and the threshold as the output, to set thresholds for both input and memory neurons. For

each masking fraction, a subset of input patterns (20%) was randomly sampled to estimate the average fraction of 1's, serving as the input for the linear functions. Specifically, for the BTSP with $n = 39,000$, we established the linear function for the memory neurons with coefficients $\alpha_1 = 4707.05$ and $\beta_1 = 10.7$ and for the memory neurons, with coefficients $\alpha_2 = 692.3$ and $\beta_2 = 26.6$; for the BTSP with $n = 25,000$, we established the linear function for the memory neurons with coefficients $\alpha_1' = 4936.7$ and $\beta_1' = 10.11$, and for the memory neurons, with coefficients $\alpha_2' = 194.64$ and $\beta_2' = 19.56$. These coefficients were obtained through preliminary simulation. Initially, we simulated various thresholds within the range [1, 100] for input and memory neurons for every masking fraction during the recall phase and recorded the $HD(\boldsymbol{x}, r(\boldsymbol{x'}))$ between each pair of thresholds. The pair of thresholds that yielded the smallest HD for each masking fraction was then utilized as data points for the function fitting.

For the binary line drawing task, we employed 50x50 pixel sparse binary line drawings as input patterns. Each pattern comprised five 8-pixel lines, randomly positioned and oriented either vertically or horizontally. In a sequence of 2,000 patterns, we included pairs of similar patterns, each sharing three overlapping lines. The task parameters were set as $n = 39,000$, $f_q = 0.005$, and $f_w = 0.6$ and the linear fitting function method was also applied to set the neuron firing thresholds for both input and memory neurons.

Details of analysis of the repulsion effect (Fig. 6). We generated 1000 orthogonal memory patterns with the input density f_p. After that, we randomly selected one pattern and generated another pattern with a 40% overlap ratio, as well as a pattern that had no overlap 1's with the chosen pattern. The generated patterns, along with other memory items, were randomly presented to the network and BTSP was applied to each pattern once. After learning, we calculated the overlapping similarity between the selected patterns and the two specific patterns separately.We also calculated the same overlapping similarity between the projection of these patterns using the random projection model. The projection results were binarized by applying the Top-K operations, ensuring that only the $\lfloor n * fq \rfloor$ neurons with the largest projection values could be activated. We set $f_q = 0.02$ in Fig.6B and employed other default parameters for the simulation. The results were averaged over 20 runs with different randomly initialized parameters.

Acknowledgments We would like to thank Jeffrey C. Magee, Aaron D. Milstein, and Sandro Romani for their helpful comments on the first version of this manuscript from (Wu and Maass, 2023), and Alice Dauphin for a critical reading of the current version. This research was partially supported by the Human Brain Project (Grant Agreement number 785907) of the European Union, and a grant from Intel.

References

Andersen, P., Morris, R., Amaral, D., Bliss, T., and O'Keefe, J. (2007). *The hippocampus book*. Oxford university press.

Bartol Jr, T. M., Bromer, C., Kinney, J., Chirillo, M. A., Bourne, J. N., Harris, K. M., and Sejnowski, T. J. (2015). Nanoconnectomic upper bound on the variability of synaptic plasticity. *elife*, 4:e10778.

Bittner, K. C., Grienberger, C., Vaidya, S. P., Milstein, A. D., Macklin, J. J., Suh, J., Tonegawa, S., and Magee, J. C. (2015). Conjunctive input processing drives feature selectivity in hippocampal ca1 neurons. *Nature neuroscience*, 18(8):1133–1142.

Bittner, K. C., Milstein, A. D., Grienberger, C., Romani, S., and Magee, J. C. (2017). Behavioral time scale synaptic plasticity underlies ca1 place fields. *Science*, 357(6355):1033–1036.

Chanales, A. J., Oza, A., Favila, S. E., and Kuhl, B. A. (2017). Overlap among spatial memories triggers repulsion of hippocampal representations. *Current Biology*, 27(15):2307–2317.

Chéreau, R., Williams, L. E., Bawa, T., and Holtmaat, A. (2022). Circuit mechanisms for cortical plasticity and learning. In *Seminars in Cell & Developmental Biology*, volume 125, pages 68–75. Elsevier.

Cone, I. and Shouval, H. Z. (2021). Behavioral time scale plasticity of place fields: Mathematical analysis. *Frontiers in computational neuroscience*, 15:640235.

Dasgupta, S., Stevens, C. F., and Navlakha, S. (2017). A neural algorithm for a fundamental computing problem. *Science*, 358(6364):793–796.

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

Grienberger, C. and Magee, J. C. (2022). Entorhinal cortex directs learning-related changes in ca1 representations. *Nature*, pages 1–9.

Guzman, S. J., Schlögl, A., Frotscher, M., and Jonas, P. (2016). Synaptic mechanisms of pattern completion in the hippocampal ca3 network. *Science*, 353(6304):1117–1123.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558.

Hopfield, J. J. (2007). Hopfield network. *Scholarpedia*, 2(5):1977.

Hu, S., Liu, Y., Liu, Z., Chen, T., Wang, J., Yu, Q., Deng, L., Yin, Y., and Hosaka, S. (2015). Associative memory realized by a reconfigurable memristive hopfield neural network. *Nature communications*, 6(1):7522.

Larkum, M. E. (2022). Are dendrites conceptually useful? *Neuroscience*, 489:4–14.

Li, P. Y. and Roxin, A. (2023). Rapid memory encoding in a recurrent network model with behavioral time scale synaptic plasticity. *bioRxiv*, pages 2023–05.

Li, Y., Briguglio, J., Romani, S., and Magee, J. C. (2023). Mechanisms of memory storage and retrieval in hippocampal area ca3. *bioRxiv*, pages 2023–05.

Magee, J. C. and Grienberger, C. (2020). Synaptic plasticity forms and functions. *Annual review of neuroscience*, 43:95–117.

Milstein, A. D., Li, Y., Bittner, K. C., Grienberger, C., Soltesz, I., Magee, J. C., and Romani, S. (2021). Bidirectional synaptic plasticity rapidly modifies hippocampal representations. *Elife*, 10:e73046.

Sugar, J. and Moser, M.-B. (2019). Episodic memory: Neuronal codes for what, where, and when. *Hippocampus*, 29(12):1190–1205.

Sun, J., Xiao, X., Yang, Q., Liu, P., and Wang, Y. (2021). Memristor-based hopfield network circuit for recognition and sequencing application. *AEU-International Journal of Electronics and Communications*, 134:153698.

Tang, S., Zhang, Y., Li, Z., Li, M., Liu, F., Jiang, H., and Lee, T. S. (2018). Large-scale two-photon imaging revealed super-sparse population codes in the v1 superficial layer of awake monkeys. *Elife*, 7:e33370.

Tarkov, M. S. (2019). Hopfield associative memory with quantized weights. In *Advances in Neural Computation, Machine Learning, and Cognitive Research II: Selected Papers from the XX International Conference on Neuroinformatics, October 8-12, 2018, Moscow, Russia*, pages 91–97. Springer.

Tsodyks, M. V. and Feigel'man, M. V. (1988). The enhanced storage capacity in neural networks with low activity level. *Europhysics Letters*, 6(2):101.

Wammes, J., Norman, K. A., and Turk-Browne, N. (2022). Increasing stimulus similarity drives nonmonotonic representational change in hippocampus. *ELife*, 11:e68344.

Wang, Y., Yu, L., Wu, S., Huang, R., and Yang, Y. (2020). Memristor-based biologically plausible memory based on discrete and continuous attractor networks for neuromorphic systems. *Advanced Intelligent Systems*, 2(3):2000001.

Wanjia, G., Favila, S. E., Kim, G., Molitor, R. J., and Kuhl, B. A. (2021). Abrupt hippocampal remapping signals resolution of memory interference. *Nature communications*, 12(1):1–11.

Widrich, M., Schäfl, B., Pavlović, M., Ramsauer, H., Gruber, L., Holzleitner, M., Brandstetter, J., Sandve, G. K., Greiff, V., Hochreiter, S., et al. (2020). Modern hopfield networks and attention for immune repertoire classification. *Advances in Neural Information Processing Systems*, 33:18832–18845.

Wu, Y. and Maass, W. (2023). Memory structure created through behavioral time scale synaptic plasticity. *bioRxiv*, pages 2023–04.

Zheng, L., Gao, Z., McAvan, A. S., Isham, E. A., and Ekstrom, A. D. (2021). Partially overlapping spatial environments trigger reinstatement in hippocampus and schema representations in prefrontal cortex. *Nature Communications*, 12(1):6231.

Supplementary Information for

One-shot learning and robust recall with BTSP, a biological synaptic plasticity rule

Yujie Wu and Wolfgang Maass†
Institute of Theoretical Computer Science, Graz University of Technology, Graz, Austria
†Corresponding author. E-mail: yujie.wu@igi.tugraz.at; maass@igi.tugraz.at

December 30, 2023

This PDF file includes

S1. Comparison of the performance of BTSP from Eq. (1) and core BTSP **2**

Figure S1. Comparison of the performance of both versions of BTSP for recall with partially masked cues 2

S2. Distribution of inputs to memory neurons for BTSP and random projections **2**

Figure S2. Comparison of the distributions of inputs to 3 further samples of memory neurons for BTSP and RP 2

S3. Comparison of recall results for perturbed and partially masked cues **3**

Figure S3. Comparison of results for recall with perturbed and partially masked cues 3

S4. Impact of f_q on recall performance of BTSP **4**

Figure S4. Comparison of the memory traces created by BTSP with reduced values of f_q 4

S5. Dependence of CAM performance with BTSP on the value of f_q **4**

Figure S5. Comparison of the input reconstruction capability of BTSP with smaller values of f_q . . . 4

S6. Scaling up the size of input patterns to the size of area CA3 in the human brain **5**

Figure S6. Scaling-up the number and sizes of input patterns 5

S7. Details to the evaluation of CAM properties of HFNs with binarized weights **6**

Figure S7. Analysis of CAM performance of a HFN with various binarization strategies for synaptic weights and different values of thresholds v_{th}^H of neurons 6

S1. Comparison of the performance of BTSP from Eq. (1) and core BTSP

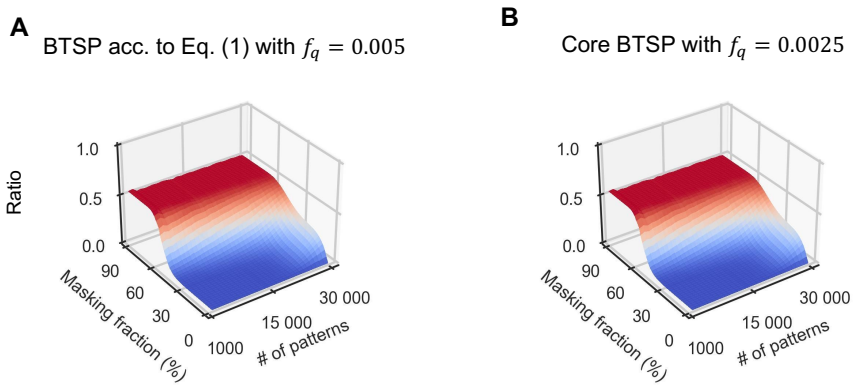

Figure S1: **Comparison of the performance of both versions of BTSP for recall with partially masked cues**. (**A**) Results for the BTSP version from Eq. (1), replotted from Fig. 3C. (**B**) Results of the same experiment with core BTSP, with $f_q = 0.0025$. Other parameter values remain the same. One sees that the performance of both versions of BTSP is virtually the same.

S2. Distribution of inputs to memory neurons for BTSP and random projections

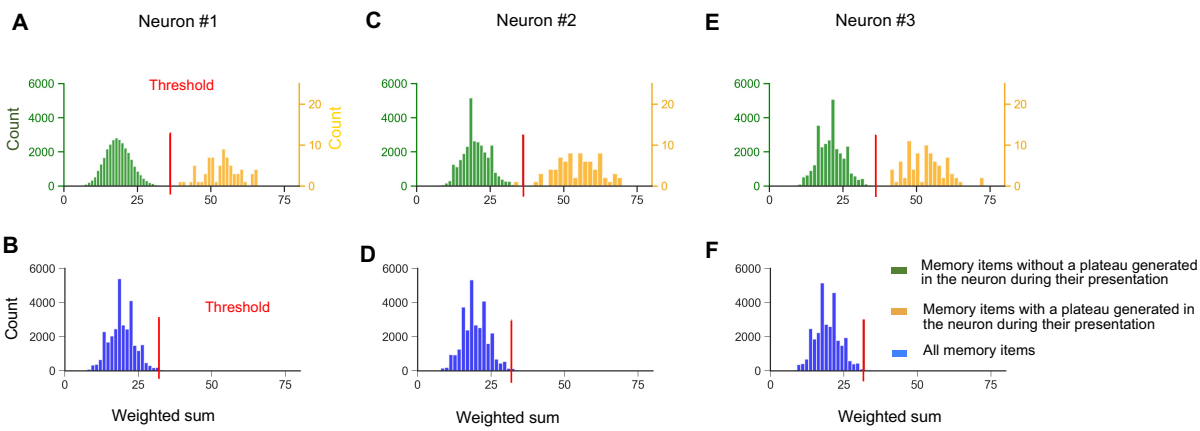

Figure S2: **Comparison of the distributions of inputs (=weighted sums) to 3 further samples of memory neurons for BTSP and RP; corresponding to Fig. 3 A, B for three further samples of memory neurons.** Panels (**A**,**C** and **E**) Same BTSP experiment as for Fig. 3 B, for 3 further randomly drawn memory neurons. Panels (**B**,**D** and **F**) Same RP experiment as for Fig. 3 A, for 3 further randomly drawn memory neurons.

S3. Comparison of recall results for perturbed and partially masked cues

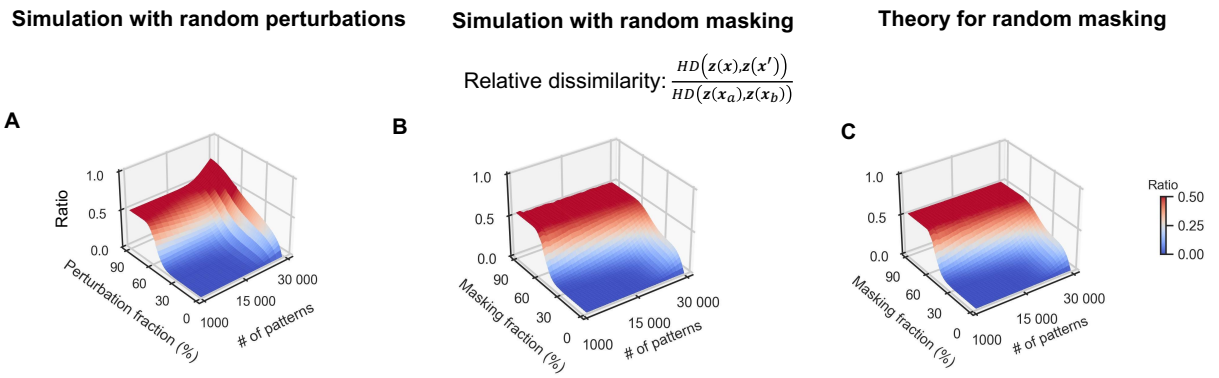

Figure S3: **Comparison of results for recall with perturbed and partially masked cues**. (**A**) Recall of memory traces with perturbed cues, where not only 1's were replaced by 0's, but also 0's were replaced by 1's; keeping however the average number of 1's the same as in the original input patterns. For any given perturbation fraction f'_d, it is assumed that each 1 in an input pattern is switched to 0 with a probability equal to f'_d, and each 0 is switched to 1 with a probability equal to $(mf'_d f_p)/(m(1-f_p)) = f'_d f_p/(1-f_p)$. The optimization process and parameter setups remain the same as in Fig. 3C, except that the thresholds were separately optimized for the two types of tests with perturbed and partially masked input patterns. The theoretical results for panel C are derived from Eqs. (17) and (21). One sees that the impact of perturbation on recall performance is virtually the same as for partially masked input patterns.

S4. Impact of f_q on recall performance of BTSP

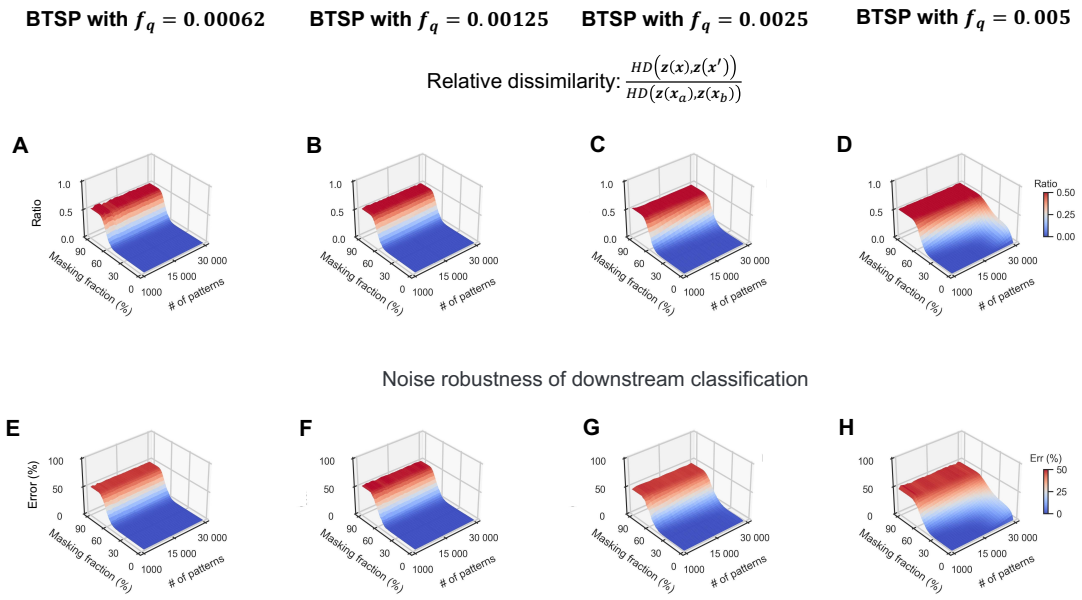

Figure S4: **Comparison of the memory traces created by BTSP with reduced values of f_q.** Panels **D**And **H** for the default value of f_q are copied from Fig.3C and F. Panels (**A**) to (**C**) show results of the same experiments but with lower values of f_q. One sees that the performance improves a little bit further for lower values of f_q.

S5. Dependence of CAM performance with BTSP on the value of f_q

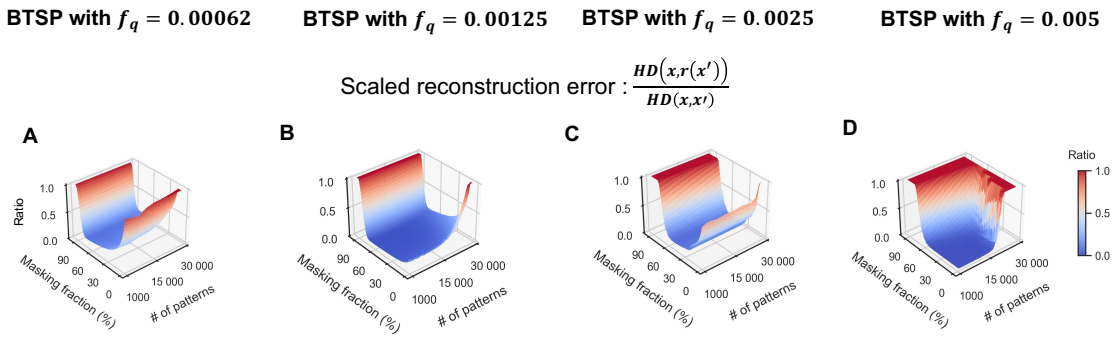

Figure S5: **Comparison of the input reconstruction capability of BTSP with smaller values of f_q.** Panel D shows again the performance for the default value of f_q, copied from Fig.5D. (**A**-**C**) Corresponding CAM performance results with lower values of f_q. One sees that the performance is improved with a lower value of f_q.

S6. Scaling up the size of input patterns to the size of area CA3 in the human brain

We have so far examined memory structures for a fixed size of the model that we introduced at the beginning, which corresponds to about 1/100 of the number of neurons in corresponding structures of the human brain. We address here the question how the memory capacity of the BTSP model changes when we use it to store 100 times larger input patterns that roughly correspond to the size of area CA3 in the human brain. Simultaneously we multiply the probability f_p that an input neuron is active for a memory item $\boldsymbol{x}$ with the factor 1/100 so that on average the same number of input neurons are active for a memory item $\boldsymbol{x}$. We keep the number n of memory neurons the same as in our default setting.

A similar scaling up of the size of input features arises when one takes a more complex neuron model into account, even if one stays at the scale of the mouse brain. The recent review (Larkum, 2022) suggests in Fig. 4 a qualitative rule for synaptic integration in a pyramidal cell with dendrites whereby each segment of its dendritic arbor should be viewed as an independent small integration zone that generates an NMDA spike if several synapses in this segment are co-active. If one assumes for simplicity that there are exactly 2 synapses in each dendritic segment, and that this segment generates an NMDA spike if and only if these two synapses are co-active, one arrives at a virtual input space that consists of all pairs of presynaptic neurons that form synapses at a common dendritic segment. This virtual input space is in general much larger. At the same time, the sparseness of activity is decreased by a corresponding factor because both synapses at a dendritic segment are more rarely co-active than individual presynaptic neurons.

We find that this change in the pattern size has a drastic impact on the memory capacity and recall capability with partial cues, see Fig. **S6**B, compared with Fig. 3C. Due to computational resource limitations, simulations were only conducted for up to 20,000 patterns. Since there is almost perfect agreement between theoretical predictions and simulation outcomes for up to 20,000 patterns, compared Fig. **S6**A and B, the theoretical predictions appear to be trustworthy also for larger numbers of input patterns. Results of these theoretical predictions for up to 800,000 input patterns are shown in Fig. **S6**B.

Fig. **S6**A and B show that in this setting with 100 times larger input patterns the attractor properties of memory traces created through BTSP become even more pronounced: Good recall is possible even when up to 2/3 of the bits of a memory item have been masked in a recall cue. Hence, both the capacity and the robustness of recall are substantially improved in this setting, which is arguably more realistic from the biological perspective. This result also suggests that one can build with the help of BTSP powerful CAM models in neuromorphic hardware that can store up to 800,000 patterns, and support robust recall even when more than 2/3 of the features of a memory item are missing in a recall cue.

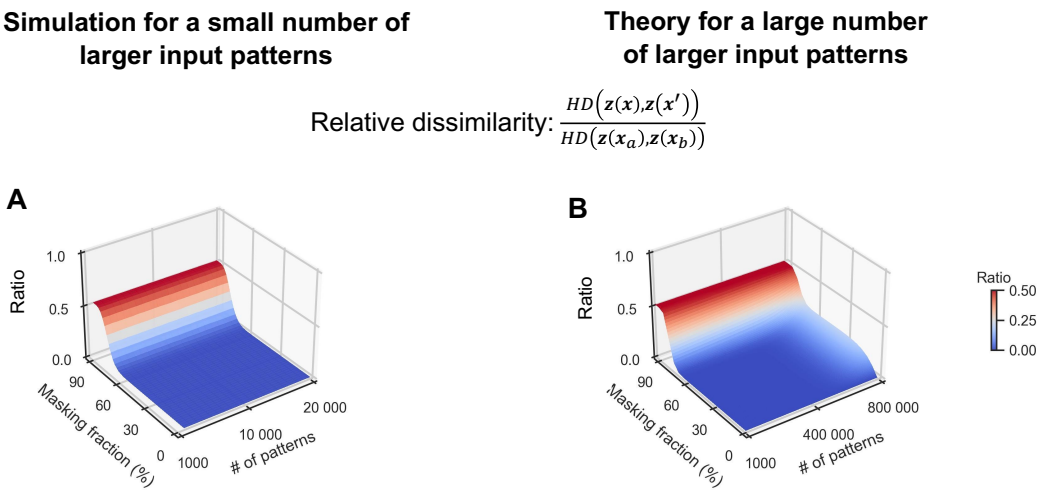

Figure S6: **Scaling-up the number and sizes of input patterns.** (**A**) Results of numerical experiments are shown for up to 20,000 input patterns that are each 100 times larger, but where the probability that an input bit has value 1 is 1/100 times smaller. (**B**) Theoretical predictions for the same type of input patterns, derived from Eqs. (17) and (21), but for up to 800,000 input patterns. One sees excellent agreement with the simulation results of A for smaller pattern numbers. These theoretical results predict very robust recall with cues where up to 2/3 of bits are masked, for up to 800,000 patterns.

.

S7. Details to the evaluation of CAM properties of HFNs with binarized weights

We employed a fixed threshold to binarize the HFN's weights. We set the weight values to 0 or $1/m$, thereby aligning the range of binarized values with that of the original weight matrix, $\boldsymbol{W}^H$, defined in Eq.(31). Three different binarization threshold values were evaluated for $\boldsymbol{W}^H$: (1) 0, (2) the median of $\boldsymbol{W}^H$, and (3) the mean of $\boldsymbol{W}^H$. For each binarization threshold, we searched for the value of v_{th}^H from $[0.0, 0.02]$ with the minimum step 0.001 with the same optimization criterion as used for the continuous weights (see *Methods*). Results for input completion with different binarization thresholds and different v_{th}^H are provided in Fig. **S7**. The resulting optimal threshold, $v_{th}^H = 0.001$, was adopted in the main text.

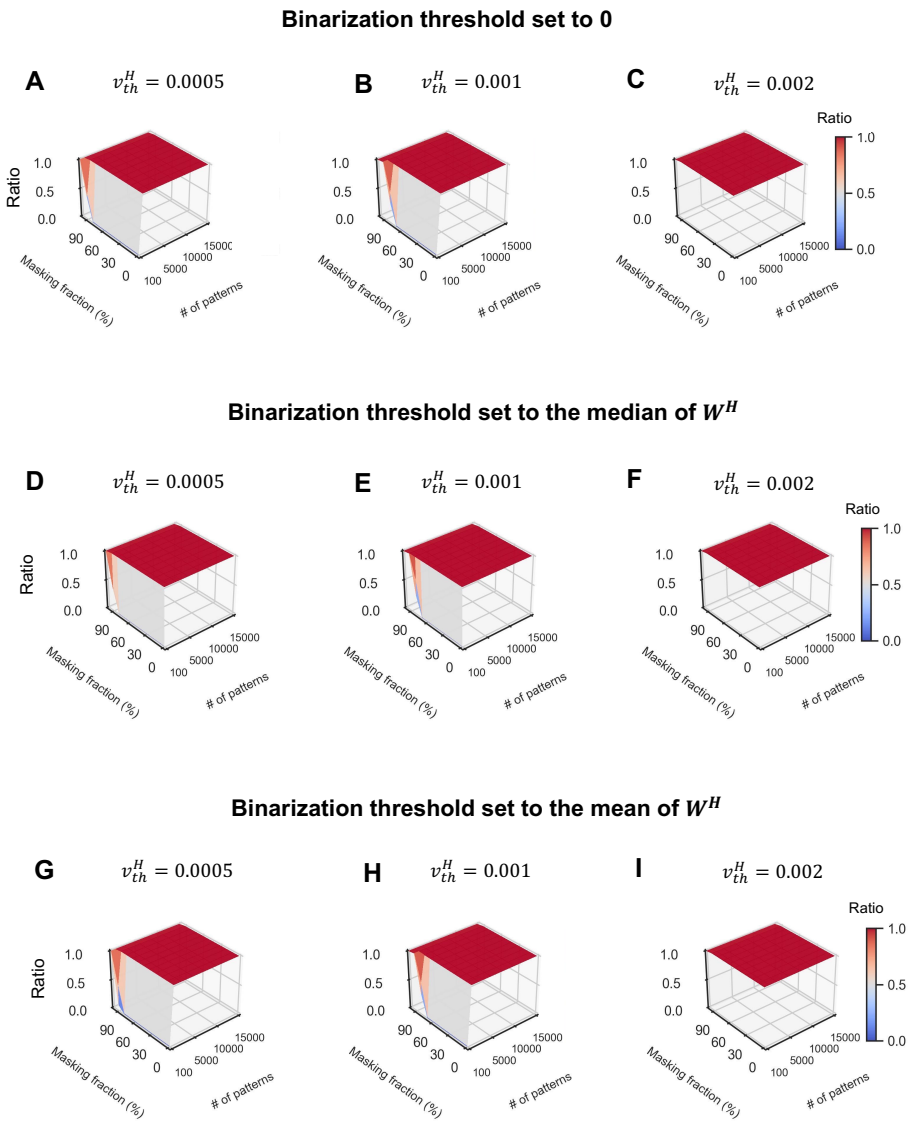

Figure S7: **Analysis of CAM performance of a HFN with various binarization strategies for synaptic weights and different values of thresholds v_{th}^H of neurons; compared with Fig.5G**. (**A**-**C**) Simulation results of HFN using a binarization threshold of 0 with different v_{th}^H values. (**D**-**F**) Simulation results of HFN using a binarization threshold equal to the median of the weight matrix $\boldsymbol{W}^H$ (defined in Eq.(31)) with various v_{th}^H values. (**G**-**I**) Simulation results of HFN using a binarization threshold equal to the mean of the weight matrix $\boldsymbol{W}^H$ with different v_{th}^H values.

References

Larkum, M. E. (2022). Are dendrites conceptually useful? *Neuroscience*, 489:4–14.