Digital computing through randomness and order in neural networks

Alexandre Pitti, Claudio Weidmann, Mathias Quoy

Digital computing through randomness and order in neural networks

Alexandre Pitti[a,1], **Claudio Weidmann**[a], **and Mathias Quoy**[a,b]

[a]ETIS laboratory, CY Cergy-Paris University, ENSEA, CNRS, UMR8051; [b]IPAL, CNRS, Singapore

This manuscript was compiled on May 29, 2022

We propose that coding and decoding in the brain are achieved through digital computation using three principles: relative ordinal coding of inputs, random connections between neurons, and belief voting. Due to randomization and despite the coarseness of the relative codes, we show that these principles are sufficient for coding and decoding sequences with error-free reconstruction. In particular, the number of neurons needed grows linearly with the size of the input repertoire growing exponentially. We illustrate our model by reconstructing sequences with repertoires on the order of a billion items. From this, we derive the Shannon equations for the capacity limit to learn and transfer information in the neural population, which is then generalized to any type of neural network. Following the maximum entropy principle of efficient coding, we show that random connections serve to decorrelate redundant information in incoming signals, creating more compact codes for neurons and therefore conveying a larger amount of information. Henceforth, despite the unreliability of the relative codes, few neurons become necessary to discriminate the original signal without error. Finally, we discuss the significance of this digital computation model regarding neurobiological findings in the brain and more generally with artificial intelligence algorithms, with a view toward a neural information theory and the design of new digital neural networks.

sparse coding | digital computing | maximum entropy | ordinal codes | continual learning | efficient codes

Because neurons represent a small energetic resource albeit with poor computational capabilities, it is expected that they rely on an efficient coding mechanism to convey maximum information (1–3). Accordingly, it is now accepted that neurons encode stimuli in a distributed fashion and transmit near independent (nonredundant and uncorrelated) information in many brain areas (4–6). For instance, distribution and sparsity help a population of neurons reduce communication errors and transmit one complete signal despite the noise in the synapses and the finite precision of the neurons. In vision, maximizing information implies forcing the coding of images into new representations in terms of the actual "primitives" of the images (4, 7), and their patterns (8, 9). These representations constitute a more compact repertoire that may well be easier to work with than a much larger redundant representation in the image. The first experiences in testing the theory of efficiency coding or redundancy reduction came from the work of Laughlin applied to the fly-eye (10). He measured and compared both the contrast distribution in the image and the contrastive cells in the fly-eye and predicted that optimal encoding would take the form of maximizing contrast by transforming the original (redundant) distribution into a uniform (uncorrelated) distribution to be transmitted to the fly brain. As each output value becomes equiprobable, the conveyed signal achieves the capacity limit for transmission with optimal bandwidth. As a result, optimal coding makes the signal resemble white noise (maximum entropy): a coding effect that is called whitening (11). Another example of efficient coding is observed in the optic nerve, which constitutes a bottleneck in transmitting information to the brain, as it comprises 1.7 million ganglion cells although the number of photoreceptors is on the order of 126 million cells. A reduced code constructed from the difference between photoreceptors in the retina (e.g., a differential code) is sent to the brain with the same amount of information, which requires far less channel capacity (11, 12).

Hence, efficient coding for pattern separation and pattern completion is hypothesized to occur widely in the brain to manipulate natural input. The same hypothesis is expected for memory access, storage, and retrieval in areas such as the hippocampus and the prefrontal cortex (6, 13). The question is, therefore, how the brain can encode, transmit and store one coherent signal efficiently from unreliable neurons. How can neurons communicate as much information as is theoretically possible? These two questions are tightly linked to the binding problem as well: how can neural areas specialize in different modalities, with partial access to information on an external event, coherently share their views to produce a unified representation of the world?

As a novel paradigm, we propose a mechanism that exploits a neural code with a limited resolution to code and decode higher resolution sequences. Resolution is the number of items within an input repertoire or the number of differentiated values that can be learned by a neuron. Therefore, high resolution denotes a small quantization step and thus

Significance Statement

We hypothesize that the brain performs some kind of digital computing to overcome the unreliability of its neurons by exploiting random synaptic connections. Theoretical equations and computer experiments show that surprisingly, only a small number of neurons is enough for coding sequences formed from items taken within a repertoire of a billion different values. Despite the learning limitation of neurons, which perform large errors, random connections help to generate high entropy, conveying maximum information simultaneously. This result led us to derive a general equation of the learning capabilities in neural networks, which has deep implications in neuroscience modeling and artificial intelligence toward developing a neural information theory.

A.P. conceived the project. A.P. implemented networks. A.P., C.W., and M.Q. performed analysis. A.P., C.W. and M.Q. contributed ideas. A.P. prepared graphics. A.P. performed experiments. A.P., C.W. and M.Q. managed the project. A.P., C.W. and M.Q. wrote the paper.

we declare no conflict of interest.

[1]To whom correspondence should be addressed. E-mail: alexandre.pitti@cyu.fr

a high repertoire cardinality (number of quantization levels). In our neural network, the neural code takes the form of a relative code to represent the relative order of items within the sequence; e.g., the following relative ordinal code *[#5, #3, #2, #4, #1, #6]* corresponds to the sequence of index *[18, 13, 8, 14, 5,19]*; see Fig. 1. Despite the coarseness of these neural codes, our results show that this mechanism can achieve error-free reconstruction by having sparse and distributed neural representations of the original sequence.

Neural cells sensitive to serial order in sequences have been found extensively in the prefrontal cortex (14–18) and the hippocampus during spatial exploration and memory tasks (19, 20). Studies identify them either as conjunctive cells or as disjunctive cells, whether researchers are looking at their binding feature to respond nonlinearly to various signals in a mixed form (21–23) or at their factorizing feature to respond only to relational or structural information (20, 24–26). In both cases, they can be viewed as loosely salient pattern detectors that are important for memory storage (27) and neural communication (28).

We devise two important and apparently antagonistic design principles in terms of information processing underlying the mechanism we propose, namely *orderliness* and *randomness*. First, relative ordinal codes permit a limited number of synapses and a limited number of synaptic weight values to represent one original item sequence. By doing so, it permits the cost of wiring (3) to be reduced and the precision of the neurons to be purposely limited; in effect, ordinal codes produce a quantized representation of one original sequence with discrete values, similar to binary codes in human-made communications networks. Second, we impose some random permutations of those ordinal codes to obtain sparse representations. Random permutations, in effect, decorrelate redundant information in incoming signals: henceforth, few neurons become necessary to discriminate the original signal. By avoiding redundancy, we guarantee the code efficiency with maximum information compression, following the maximum entropy principle (8).

In our neural network, although any individual neuron is unreliable in reconstructing an original signal due to the coarse quantization performed by the relative order codes (pattern separation), taken at the population level, they can share their views and exploit their distributivity and sparsity to find a consensus through crosstalk (pattern completion). Such an error-correction process can be performed iteratively by selecting at each time step the solutions that best satisfy the constraints during reconstruction. This approach can be seen as similar to a Bayesian treatment of information using conditional probabilities, such as expectation-maximization, active inference (29, 30), predictive coding (31–33) or free-energy minimization (34–36); see Fig. 2.

In effect, our strategy emphasizes the role of using simple decoders to achieve noise cancellation. Interestingly, the three computational stages we describe, namely, (1) shuffling, (2) low-resolution codes, and (3) belief voting, are reminiscent of the error-correcting mechanisms employed in modern digital communications pioneered with the turbo-codes invented by Claude Berrou and Alain Glavieux (37). Turbo codes exploit two or more differently shuffled versions of the input signal encoded by simple codes. The weakness of these codes causes large individual reconstruction errors. However, iterative combinations of their belief votes allows us to find a consensus and to correct errors near perfectly. Shannon showed that communication channel capacity can be attained for asymptotically long messages using a random channel code, which maps each message to a codeword that realizes of independent identically distributed symbol random variables (38). However, such a code is impractical in terms of complexity. It was the advent of turbo codes that first demonstrated the existence of structured codes operating close to capacity with reasonable decoding complexity (39). Turbo codes encode a message twice using a simple code, once in original positional order and once in randomly shuffled order. These two "views" on the message allow for an elaborate iterative decoding algorithm that corrects many more errors than the simple codes alone.

In agreement with this, we suggest that random shuffling and ordinal neurons can embody efficient error-correcting codes to represent information, thanks to their sparsity/distributivity features and the consensus at the neural population level, despite their limited learning capabilities due to a discrete representation and crude synaptic resolution. This mechanism has the advantage of making memory *digital* because it encodes information using a set of discrete values, making it more robust to intrinsic and external noise for memory preservation and retrieval.

In our experiments, we found that a surprisingly small number of neurons (a few dozen) is enough to encode and decode a sequence of distinct items taken in a large repertoire and this number grows linearly while the repertoire size can grow exponentially (e.g., order of a billion of symbols). This is a novel and fundamental result: it demonstrates that the number of neurons needed to learn an input is related to its resolution (which also determines its entropy) and to the resolution of the neural codes. To our knowledge, this observation has never been made before, it expresses that the learning capacity of a neural network is limited by its entropy *and* the entropy of the object to be learned. It closely follows Shannon's source and channel coding theorems and provides insights for a neural information theory.

Our contributions in this paper are, therefore, twofold. First, we describe a novel type of neural network, a *digital neural network*, and its design. Second, from this discovery, we could derive the Shannon equations for its capacity limit to learn and transfer information; see Eqs. 1 and 2 in the *SI* section. These equations are however universal and can be extended now to any sort of formal model of neural networks (e.g., deep networks, spiking networks, reservoir networks) to define and analyze their learning capacity limit. Similar to digital processing using binary bits in human communication and in memory storage devices, the possibility of having digital neural networks is a direct consequence of this equation. To our knowledge, these are two novel results that were exploited neither in artificial intelligence nor in computational neuroscience.

We discuss the impact of our findings on the brain's memory organization (perception, recall, binding), its implications in terms of neural coding for sparsity and distributivity, and on the future design of energy-efficient and large-scale artificial intelligence systems.

Model

Glossary. We explain some (information-theoretic) terms used in the following.

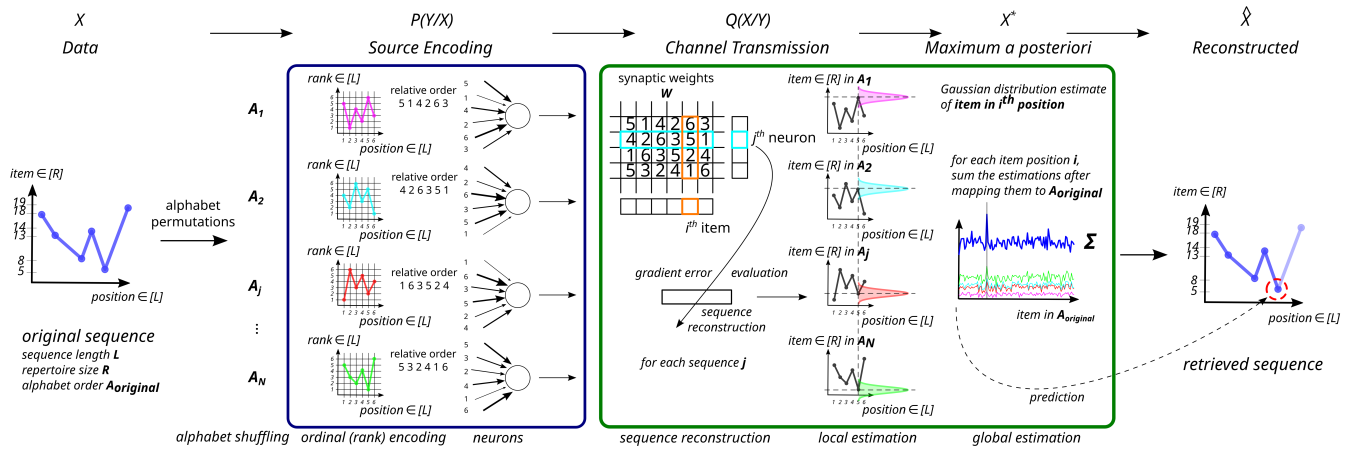

Fig. 1. Schematic presentation of the neural population based on randomly permuted ordinal codes. The process has three stages: the encoding of the original sequence, its decoding and a global belief vote. In the first phase, the neurons encode the relative order (the ranks) of the items in the spatiotemporal sequence X using multiple randomly shuffled orderings of the item alphabet. The result is that each neuron sees a randomly permuted ordinal code, e.g., $P(Y/X)$. The items' values are no longer present in the ordinal codes, which perform a drastic quantization of information. During the decoding phase, each neuron reconstructs the sequence in its alphabet ordering by trial and error, e.g., $Q(X/Y)$. Thus, each neuron has a different *local* estimate of the items in the sequence. In the final stage, after mapping back the local alphabet orderings to the original, a *global* belief vote at the population level accumulates the local decisions from all the neurons, allowing correction of local decision errors; e.g., maximum a posteriori probability $\hat{X} = X^*$.

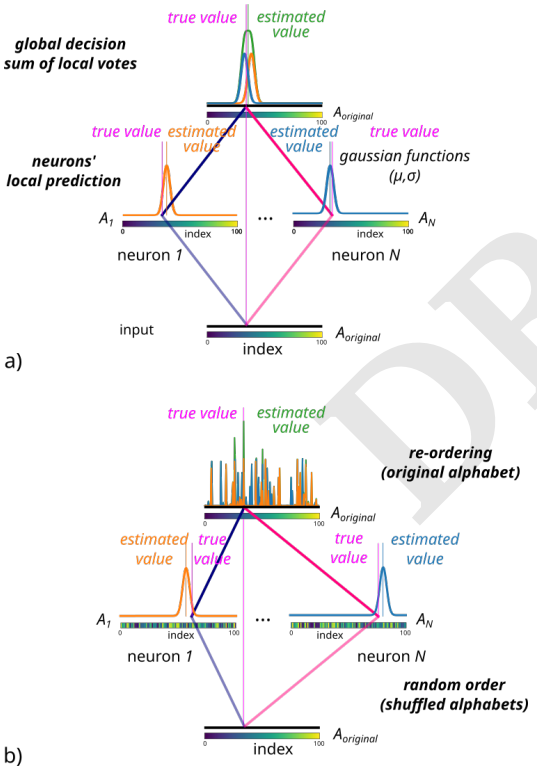

Fig. 2. Robust neural decoding of a signal encoded with and without random permutation of the input repertoire (alphabet). a) Encoding using N neurons *without* permuting the alphabet order. Local (per neuron) errors are modeled by a Gaussian distribution. Combining the N local estimates (top) allows only for a linear reduction in the estimation noise through averaging. b) Random alphabet permutations cause repertoire items that are neighbors in the original order to lie farther apart. When cumulating the reordered local Gaussian votes (top), this leads to a nonlinear effect that lets the global estimate stand out in the noise, which is now spread over the entire alphabet.

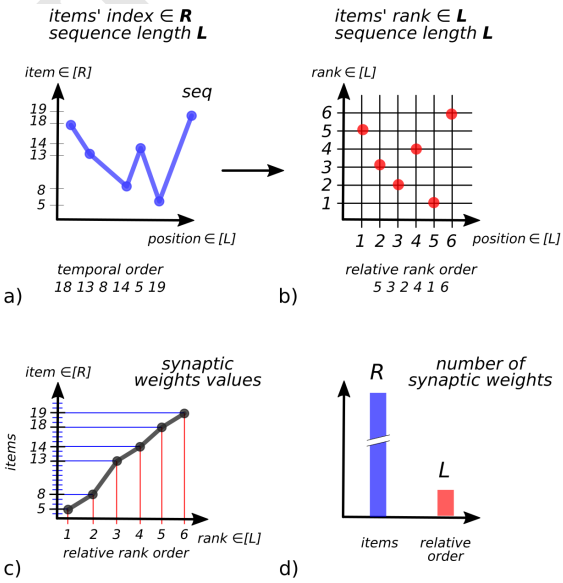

Fig. 3. Efficiency of a relative order code versus a temporal code. a) A spatiotemporal sequence of L items taken in a repertoire of size R. b) Ordinal codes represent the sequence with a vector from L only, storing the relative rank order over time of the items in the sequence. In c-d), in terms of computational cost and precision, formal neurons such as perceptrons have to encode the items' index of temporal sequences either in their synaptic weights with resolution R and L synaptic links or with R synaptic links and synaptic weights with resolution L. In c), instead, ordinal codes represent in their weights the relative order of items in the sequence only (L values). In d), this second type of coding allows the drastic quantization of information to only L synaptic links to learn, with respect to the R links necessary instead, as in formal neurons. This large reduction in dimensionality comes at the cost of losing information about the items' values.

Input sequence (message): a sequence (vector) of items that shall be encoded. The index may be related to time, position, etc. Additionally, termed message in information theory.

Item: an input value, e.g., a stimulus level, taken in a fixed repertoire.

Repertoire (alphabet): the set of possible values, assumed here to be finite and linearly ordered, e.g., the English alpha-

bet with alphabetic order, or brightness levels $\{0, 1 \dots 255\}$, with integer order.

Resolution: the quantization step size used when representing a continuous quantity (e.g., the pitch of a sound). A finer quantization resolution will then correspond to more steps, a larger repertoire, needed to cover a fixed input range.

Channel: a mapping function that models a transmission (or storage) system. If the channel map $h(\cdot)$ is one-to-one, the channel is *noiseless*, and the input may be reconstructed perfectly from its output. Otherwise, the channel is *noisy* and perfect reconstruction is impossible. Noisy channels often have probabilistic maps $h(\cdot)$.

Channel encoder: mapping of a message to a *codeword*, a sequence of channel input values. Its goal is to add structured redundancy to protect against channel noise.

Decoder: exploits the channel code structure to correct errors introduced by the noisy channel.

Digital computing: the main difference w.r.t. analog computing is that values involved in computation are from a finite set (e.g., binary in most digital computers). This requires the ability to correct the errors introduced by every type of physical computing and storage circuit (biological or electronic). The ordinal code with a belief voting decoder used in this work is an error correcting mechanism. **Turbo coding:** a telecommunications error correction scheme that encodes the original input sequence and an interleaved (position-permuted) copy of it. The decoder iteratively estimates the original sequence, aided by the previous estimate of the interleaved sequence, and vice versa. This alternating decoding iteration allows many more errors to be corrected than with a single code, provided that the interleaver ensures that the two estimates for a symbol are almost independent.

Overview. The overall process consists of encoding an information sequence, storing (transmitting) it in order-sensitive neurons, then decoding it.

The input sequence of length L consists of items taken from a repertoire of size R, which may be mapped without loss of generality to the set of integers $[R] = \{1, 2, \dots R\}$. We will interchangeably use the terms repertoire and *original alphabet*, to distinguish it from the reordered (permuted) alphabets underlying the coding and decoding mechanisms.

A message sequence $\boldsymbol{s}$ comprising L items (symbols) $\boldsymbol{s} = [s_1, s_2, \dots, s_L]$ is encoded by a function $\boldsymbol{x} = g(\boldsymbol{s})$ into a codeword $\boldsymbol{x} = [x_1, x_2, \dots, x_N]$, where the number of message symbols L and code symbols N are not necessarily equal. The codeword is transmitted through a communication (or storage) channel to produce an output $\boldsymbol{y} = [y_1, y_2, \dots, y_N] = h(\boldsymbol{x})$, which is decoded to produce a message estimate $\hat{\boldsymbol{s}}$. Message symbols, channel inputs and outputs are modeled as random variables, indicated by capital letters, while realizations use lowercase letters.

The message is channel-encoded by mapping it to N randomly permuted alphabets, which results in different rank sequences "seen" by the ordinal neurons. The latter form the channel, which outputs N scalars (dot products); see Eq. 1 below. They are two sources of channel "noise": the alphabet quantization due to the replacement of items by their rank, and possibly a many-to-one mapping of rank sequences to a scalar dot product. Finally, decoding is performed by first estimating the N rank sequences, and then combining those into a global decision vote. The details of this process are explained in the following.

Computational features of ordinal codes. Neurons sensitive to the ordinal structure within a sequence can be implemented by weighting the relative order of the items depending on their relative rank or their relative importance within it. Hence, this type of coding differs from the temporal coding of the bioinspired spike timing-dependent plasticity reinforcement rule (40, 41). Relative ordinal codes depart also from the rank-order coding algorithm of Simon Thorpe and colleagues (42, 43), although this work and previous works are inspired by it (36, 44). The rank-order coding (ROC) algorithm has been proposed as a model to explain the rapid processing performed by the visual system in a few cortical layers. This algorithm is a computational model of the visual spiking neurons and the STDP mechanism. Although sparse, the encoding is not relative; therefore, there is no reduction in the number of synaptic weights. We will develop more in the Discussion section on how the brain might carry out this function and the biological plausibility of it.

We can use the example presented in Fig. 3 a) and b) to explain the difference between the two types of coding, respectively STDP/ROC and ordinal encoding. For instance, in a time series of 6 items ordered as follows $seq : [18, 13, 8, 14, 5, 19]$ in Fig. 3 a), the ordinal code corresponding to this sequence is $order : [\#5, \#3, \#2, \#4, \#1, \#6]$ in Fig. 3 b). If R is defined as the cardinality of the input space, and L is the number of items within the sequence, then it is necessary to have L or R synaptic links to encode the sequence, depending on the code used and the desired precision level (see Fig. 3 c).

For instance, the STDP learning mechanism requires R synaptic links to encode the indices and their location in the temporal sequence; the values of the synaptic weights encode the temporal delay or the index order (40, 45, 46). In comparison, an ordinal code requires only a vector of L weights, in which the amplitude level encodes the relative order. This relative code can be seen as a harsh analog-to-digital conversion in which the exact item values in the sequence are removed. However, in the case where $R \gg L$, it can represent a computational advantage to represent only the ordinal structure within the data (47, 48), see Fig. 3 d).

Ordinal codes implementation. The ordinal coding strategy consists of discretizing the items in the sequence based on their rank in a given alphabet.

The ordering function $\text{rank}(A_n, \boldsymbol{S}, i)$, $n \in [N]$, $i \in [L]$, specifies as output the rank under order A_n of the item s_i located at position i within the sequence $\boldsymbol{S} = [S_1, S_2, \dots, S_L]$. The ordered alphabet $A_n = [\pi_1^{(n)}, \pi_2^{(n)}, \dots, \pi_R^{(n)}]$ is a permutation of the original repertoire, and N is the number of output neurons, equal to the number of representations of the same sequence in different permuted orders. We implement the rank function $\text{rank}(A_n, \boldsymbol{S}, i) = 1/r$ as the inverse of the rank r for a particular index i, which can be obtained easily with the `argsort()` function in the C, MATLAB, or python languages.

The equations of the neurons Y sensitive to ordinal information in a sequence are as follows. The neurons' output Y is computed by forming the dot product between the ordering function $\text{rank}(A_n, \boldsymbol{S}, i)$ and the synaptic weights w_i; $w_i \in [0, 1]$, $i \in [L]$. For an input sequence of L items taken in the repertoire of cardinality R and for a population of N

ordinal neurons, we have:

$$Y^{(n)} = \sum_{i=1}^{L} \text{rank}(A_n, \boldsymbol{S}, i)\, w_i^{(n)}, \quad n \in [N]. \qquad [1]$$

The updating rule of the weights is that of the Kohonen networks (49) with a learning rate α fixed to 1.0 for one-shot learning, for the neuron $Y^{(n)}$, we have:

$$\Delta w^{(n)} \quad = \quad \alpha(\text{rank}(A_n, \boldsymbol{S}) - w^{(n)}). \qquad [2]$$

Thus after complete learning, the weights $w^{(n)} = \text{rank}(A_n, \boldsymbol{S})$ and the neuron's output becomes maximal, $Y^{(n)} = Y_{\max} = \sum_{r=1}^{L} \frac{1}{r^2}$ for our choice of rank function. Notice that this maximum depends only on the choice of rank function and the sequence length L.

Since the weights w are normalized between $[0, 1]$ and the support of their density is bounded, the ordinal neurons are similar to radial basis functions. This attribute permits us to use the neurons Y as receptive fields and radial basis functions so that sequences with the same item order will fire the Y neurons with a high activity level with respect to the alphabet orders A_n. The channel input X does not appear in Eq. 1, since the chain *encoder–channel* may be seen as mapping the input sequence $\boldsymbol{S}$ directly to the channel output $Y^{(n)}$.

This neural network differs from other random neural networks, such as reservoir computing (50, 51) or sparse coding with random projections (52). In those networks, random projections tag the input in a high-dimensional space, and they cannot retrieve it. Our neural network, instead, can reconstruct the incoming signal. The random permutations decorrelate information and remove redundancy to have independent identically distributed random variables.

Sequence reconstruction mechanism. To reconstruct each sequence learned by each neuron and to have an estimate of the items in each alphabet, we can use any metaheuristic methods, such as the hill-climbing algorithm, simulated annealing, or genetic algorithms, to evaluate it by trial and error. Here, we implement the simulated annealing algorithm solutions through an iterative stochastic optimization, as the one used in related works on predictive coding (36, 44), Figure 1.

Using a noise vector $\boldsymbol{S}\_noise$, we test a generated sequence $\boldsymbol{S} = \boldsymbol{S} + \boldsymbol{S}\_noise$, and indirectly evaluate its distance to the encoded sequence $\boldsymbol{S}^*$ in neuron Y by computing the error E between the current activity level of neuron Y for sequence $\boldsymbol{S}$ and its highest activity level Y^* for the encoded sequence $\boldsymbol{S}^*$. If the error gradient ΔE diminishes, then we store the current sequence $\boldsymbol{S}^*$.

This error signal E, similar to a gradient descent or hill-climbing mechanism, guides the exploration process iteratively to minimize error until convergence.

Similar with what we showed in (36, 44) this variational process is an online stochastic hill-climbing algorithm performed iteratively; a pseudocode is provided thereinafter. We added in (47) a more sophisticated hill-climbing algorithm corresponding to simulated annealing to efficiently drive the exploration process.

Global decision vote mechanism. The global decision vote mechanism has some similarities to a Gaussian mixture model (GMM); see the pseudocode.

For each location in the sequence, a probability density function for each item in the repertoire is produced, represented as a sum of all densities of Gaussian components from all the neurons.

The N Gaussian distributions are centered on the L retrieved items for each location in the sequence *but* on their respective alphabet A_n. Therefore, neighboring items will not be the same in each alphabet and the vote will be orthogonal, as presented in Fig. 2. This differs from the original GMM.

For kernel-based methods, the parameter σ corresponds to the bandwidth parameter or the variance, which has an incidence on the global decision vote. The smaller σ is, the larger the bias, whereas the larger σ is, the smaller the bias.

In GMM, an optimal bandwidth can be defined with respect to the number of Gaussian functions, although this was not considered here.

Algorithm 1 Pseudo-code of the algorithm

$\boldsymbol{s} = [item_1, item_2, \ldots, item_L]$, ▷ a sequence of L items,
items $\in [R] = \{1, 2, \ldots R\}$ ▷ items randomly selected
neurons $\in [N]$ ▷ neural population of N neurons
random alphabets $\boldsymbol{A} = [\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_N]$, ▷ of cardinality R
original alphabet $\boldsymbol{A_0} = [1, 2, \ldots R]$

$\boldsymbol{s}_k = \boldsymbol{A}_k[\boldsymbol{s}]$, $k \in [N]$▷ sequence $\boldsymbol{s}$ in the new alphabet $\boldsymbol{A}_k$

#1 encoding, one-shot learning for demonstration purpose
for $k = 1, 2, \ldots, N$ **do** ▷ for each neuron k
  $W_k = \text{rank}(\boldsymbol{A}_k, \boldsymbol{s}_k)$ ▷ learn the relative ordinal code

#2 decoding, similar with a Hill-Climbing gradient error
for $k = 1, 2, \ldots, N$ **do** ▷ for each neuron k
  initialize Err_k, Err_bak,
  $\boldsymbol{s}\_bak = \boldsymbol{s}\_noise$ ▷ with $\boldsymbol{s}\_noise \in [R]^L$
  while $Err_k \neq 0$ **do**
    $\boldsymbol{s}'_k = \boldsymbol{s}\_bak + \boldsymbol{s}\_noise$ ▷ with $\boldsymbol{s}\_noise \in [R]^L$
    $Y^{(k)} = \sum \text{rank}(\boldsymbol{A}_k, \boldsymbol{s}'_k)\, W_k$,
    $Err_k = (Y^{max} - Y^{(k)})^2$
    if $Err_k \leq Err\_bak$ **then** ▷ keep values
      $\boldsymbol{s}\_bak = \boldsymbol{s}'_k$
      $Err\_bak = Err\_k$
  $\boldsymbol{s}_k = \boldsymbol{s}\_bak$

#3 global decision, similar to a Gaussian Mixture Model
initialize σ, $\boldsymbol{S}'$
for $i = 1, 2, \ldots, L$ **do**
  initialize $cumul\_sum[i, j] = 0$, $\forall j \in [R]$
  for $k = 1, 2, \ldots, N$ **do**
    initialize $\mu = \boldsymbol{s}'_k[i]$
    for $j = 1, 2, \ldots, R$ **do** ▷ or j in a range around μ
      $G(\pi_j^{(k)}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(j-\mu)^2/2\sigma^2}$ ▷ in alphabet $\boldsymbol{A}_k$
      $cumul\_sum[i, j] += G(j)$ ▷ in alphabet $\boldsymbol{A}_0$
  $\boldsymbol{S}'[i] = argmax(cumul\_sum[i, :])$ ▷ return max item
return $\boldsymbol{S}'$

Results

Because each neuron encodes a quantized version of the original sequence, the reconstructed sequences are prone to local errors. However, these errors are also distributed uniformly with respect to the randomization performed on the original alphabet (e.g., the original order of the items in the repertoire); see the Materials and Methods section for a glossary of the terms used. We can exploit this property to disambiguate what is information from what is noise when a decision is made at the population level. We display this effect in Fig. 4 a-d) for four reconstructed sequences with different randomly permuted alphabets for neurons 1, 2, 5 and 8.

As neurons tend to satisfy a minimal global error during reconstruction using only permuted codes, the local errors can be large in some locations. The density distribution of the local error is now uniform due to the different permuted orders for each neuron. This can be exploited for disambiguation during the global decision vote at the population level when the sequences are remapped into the original alphabet, as seen in Fig. 2 b).

This equalization of the distributions is similar to the whitening effect found in sparse coding (9, 11) and differs from conventional neural networks, whose neurons encode inputs using the same distribution and same lexicographic order as in Fig.2 a); in this case, the error distribution is correlated and blurred, and, therefore, difficult to discriminate.

To understand how the global decision is made at the population level, we analyze the local votes and their cumulative sum in Fig. 5 to reconstruct one item, taken as an example. Fig. 5 a) presents the vote between 0 and 1 of each neuron, for each alphabet item (of $R = 100$ items) and for a given position in the sequence. The local vote for each neuron follows a wide Gaussian curve centered on the retrieved item and its surroundings controlled by the bandwidth parameter σ. The votes of the different neurons are then mapped back to the original alphabet order, except for neuron 49 (last column), which has an unshuffled alphabet, to ease method understanding.

Fig. 5 b) shows the cumulative sum of each neuron's vote for each item $\in [R]$ (the sum is below 50, the size of the neural population), displaying the global decision vote at the population level, with a peak around the item of index 23, the ground truth displayed with a red arrow. This peak is also observed in Fig. 5 c), in which we overlap the cumulative sums. In addition, we can observe other peaks of smaller amplitude, which are other alternatives. The remaining majority of the votes form a noninformative background noise; i.e., the noninformative votes vanish. We can apprehend now the beneficial effect of randomization on reconstruction. Randomization permits crossing the local decisions, so that at the global level, only the intersecting votes remain.

Another question is to assess the influence of the global decision mechanism on the belief vote. To understand this, we plot in Fig. 6 the reconstruction error with respect to the cumulative local votes for each neuron and different width parameter σ. For instance, this parameter σ modifies the Gaussian functions' width, which acts upon the shape of the density distribution of the global vote when they are summed up.

The cases for $\sigma = 1, 2$ and 5 correspond to the most unreliable situations when the redundancy of the neurons is little exploited and when the error is only slowly reduced; exploiting

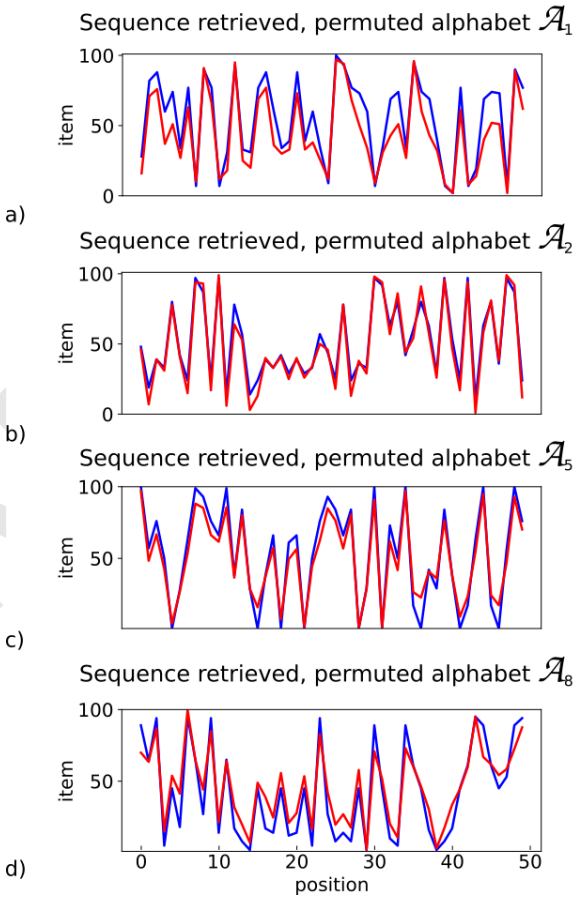

Fig. 4. Examples of reconstructed sequences with different permuted alphabets or keys, with $R = 100$ and $L = 50$. The permuted ordinal code learned by each neuron allows to retrieve the sequence with high fidelity, but always with some small local error due to the quantization to ranks performed by the neurons in their respective alphabet order. The original sequence (in the permuted order alphabet A_i of the neuron i) is plotted in blue and the retrieved sequence is plotted in red. Variance is proportional to R, approx. $\pm 0.1R$.

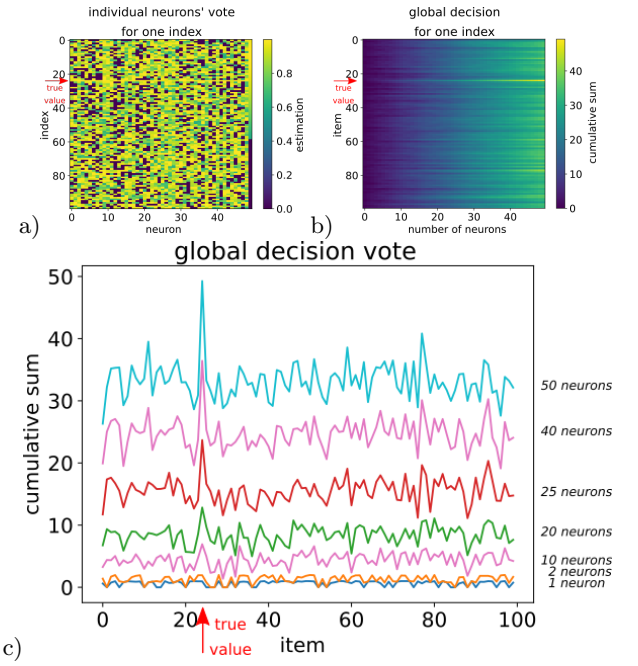

Fig. 5. Local decision vote for individual neurons and global decision vote at the neural population level; $R = 100$, $L = 50$. The red arrow indicates the true value to be retrieved back. In a), the activity level represents the local decision vote for each neuron (X axis) based on the Gaussian density distribution centered on the estimated values for each item in their respective randomized alphabet (Y axis). In b), the figure presents the cumulative sum with respect to the number of neurons used during global decision. The activity level indicates the accumulated sum for each item, the global decision vote at the neural population level. In c), display of the cumulative sum for several numbers of neurons used.

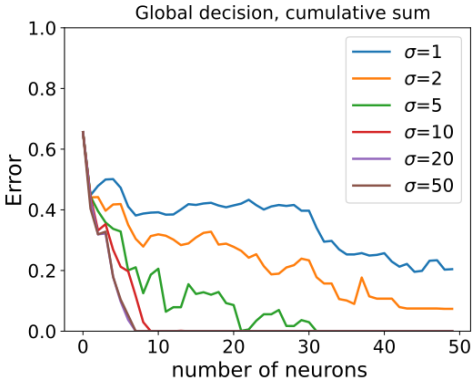

Fig. 6. Plot of the root mean-squared error (RMSE) for the global decision at the neural population level with respect to the parameter $\sigma \in [1, 5, 10, 20, 50]$; with $R = 100$ and $L = 50$. The smaller the parameter σ is, the less effective the decision-making, which will not make good use of the redundancy. In such cases, ten neurons are not enough to retrieve the original sequence, limited to an RMSE of 0.3. Instead, for a larger parameter σ above 10, fewer neurons can drastically reduce error to zero, performing a sparse coding of the incoming sequence.

the fifty neurons can reduce the overall root mean square error (RMSE) from 0.5 to 0.3 for $\sigma = 1$ and $RMSE = 0.1$ for $\sigma = 2$. However, during the decision-making process, any large classification errors (relative index errors above σ) are unlikely to be corrected when reordered. This is because the votes are centered only on the locally retrieved items. For a slightly higher value $\sigma = 5$, which corresponds to a relaxation of the decision process allowing a local error of 5 index distances from the correct index in the repertoire R, the RMSE is cancelled and only 22 neurons are necessary to reconstruct the original sequence (green line). We can remark that more neurons generate errors, and above 32 neurons, there is a complete error cancellation. In comparison, for higher values $\sigma > 10$, which correspond to a relaxation of the decision process for possibly large classification errors above a relative distance of 10 indices to the correct index, the number of neurons needed to cancel the error is reduced drastically to seven or eight neurons needed to retrieve the original sequence perfectly. These results show that an utterly small number of noisy neurons, seven or eight, can perfectly represent a sequence of fifty items taken in a repertoire of one hundred items, which can be letters or pixel values. However, above this small number, the neural population becomes redundant and more neurons will not add more information. Therefore, seven or eight neurons are the lower limit (N_{limit}) until pattern completion in sparse coding for a sequence of fifty items ($L = 50$), in a repertoire of one hundred items ($R = 100$) in this case.

We illustrate the reconstruction process and the pattern completion stage in a sequence of fifty items ($L = 50$) for a repertoire of now ten million items ($R = 10^7$) in Fig. 7. Fig. 7 a) shows the global decision vote based on the cumulative sum for different neural population sizes N. Fig. 7 b) shows the absolute error between the true items and their local reconstruction (after global consensus voting) for each position in the sequence. The consensus vote among the neurons with complete disambiguation is achieved for approximately seventeen neurons; $N = 17$. Below this limit, not all items in the sequence are correctly reconstructed. We can observe that the discrepancy is not linear and that errors cannot be predicted monotonically with respect to the size of the neural population used during the decision-making process. The error cancellation changes abruptly and nonlinearly at different locations in the sequence with respect to the number of cumulative votes.

We display (in Fig. 8 a-b) the performances of the neural population when encoding a sequence of fixed length $L = 50$ and for different cardinalities R of the input repertoire, up to one hundred million items, $R = 10^8$. Such a large cardinality may model a finely quantized fixed-range signal, so in the following, we speak interchangeably of resolution and cardinality. High resolution means a small quantization step and thus a high repertoire cardinality (number of quantization levels).

The case for input repertoire cardinality $R = 10^2$ (blue line) corresponds to the same situation as in the previous experiments.

For a sequence of items taken in a ten times larger repertoire, $R = 10^3$ (orange line), corresponding to a higher resolution, the performances degrade slightly, and only approximately ten neurons are required to fully reconstruct the sequence. That is, three additional neurons are necessary to encode a sequence taken in an alphabet of ten times as many items, which is a surprising result.

However, for higher cardinalities up to $R = 10^8$, the performances do not degrade and the progression becomes slightly more nonlinear. The number of ordinal neurons necessary to fully encode a sequence taken in a large-scale repertoire is extremely small, below twenty neurons, in comparison to the repertoire cardinality. The graph in Fig. 8 b), with the values

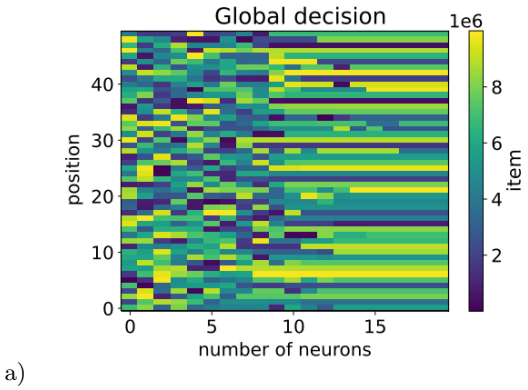

Fig. 7. Sequence reconstruction vs. number of neurons N for a fixed input sequence of length $L = 50$ (repertoire size $R = 10^7$). Each column in the matrix plot corresponds to the reconstruction for a given N. a) shows the global reconstruction in the color-coded repertoire. b) The squared reconstruction error averaged over the neurons. Approximately 17 neurons are needed to guarantee correct reconstruction in all sequence positions.

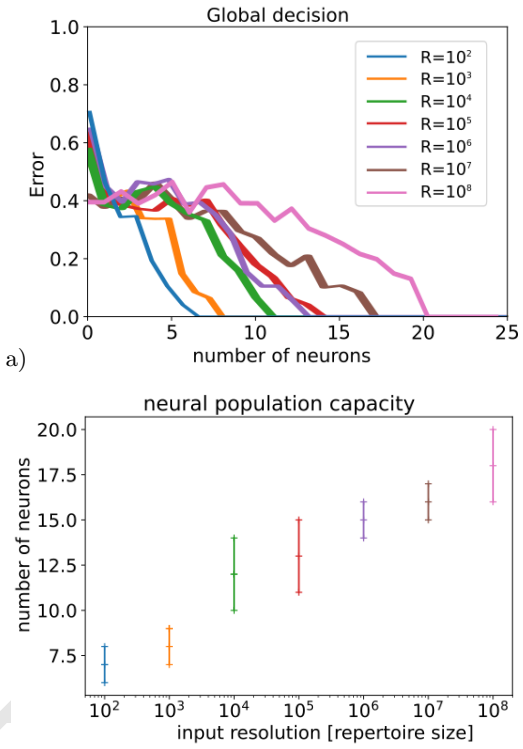

Fig. 8. Number of neurons needed to decode a sequence of various resolutions. The resolution is related to the size R of the input repertoire, or its cardinality, from which items are taken in the sequence (larger R for finer resolution). The number of neurons N_{limit} is the minimum number of neurons found necessary for reconstruction without error. In a), global decision vote for various cardinalities R; we set σ to the large value $R/2$. In b), the minimum number of neurons needed is N_{limit} to reconstruct the sequence without error with respect to the input resolution (resp. R). The graph shows a linear progression of the number of neurons required to code a sequence while the cardinality R augments exponentially; the values are averaged over ten simulations.

averaged over ten simulations shows the counterintuitive result that a logarithmic relationship can be achieved between the number of neurons necessary to encode a sequence and its resolution, which can grow exponentially.

This shows the surprising result that an extremely small number of neurons is required to disentangle a signal or sequence of very high resolution, and this is achieved with quantized neurons with limited learning capabilities, which make large reconstruction errors.

This result indicates that few quantized neurons can handle very high-resolution signals as sparse codes to keep memory safe for reduced neural communication (3) and error-free pattern completion (9, 13, 53). Put into an equation format, this fundamental result describes the intuition behind information theory that the channel capacity is related to the resolution of the message to be learned. In the case of a neural network, the number of neurons N and the resolution of their codes $R_{network}$ represent the neurons' capacity to learn one input and $R_{input} = R$, its resolution. We develop in the SI section the demonstration of the equation that relates these different terms and corresponds to: $NL \log R_{network} \geq L \log R_{input}$. This equation can be adapted to our ordinal neural network with $R_{network} = L$ being the number of possible values that can take the synapses; see Eqs. 1 and 2 in the SI section.

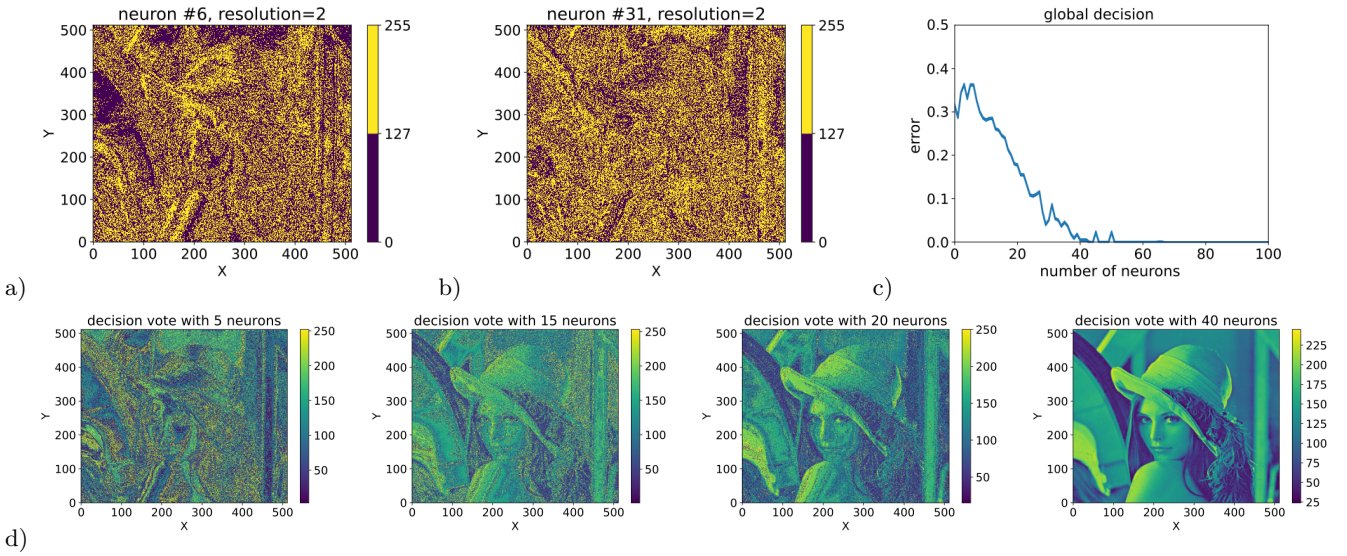

Fig. 9. Image reconstruction by neurons of lower resolution. In a), we present two neurons that encode an image with random permutations of the pixels' distribution (255 values) and reduced to a binary code (2 values). c) Euclidean error with respect to the number of neurons used during global decision-making. Near forty binary neurons are required to retrieve back perfectly the original pixels' value. d) image reconstruction for different number of neurons used during the decision vote.

We illustrate our algorithm on a visual example. The network learns one image of size 256x256 with neurons of synaptic resolution 2; see Fig. 9. Each neuron has access to a different shuffled order of the pixel values within the image repertoire of size 256. The binary codes reduce the pixel values to the values [0,1] for the synaptic weights. This operation makes neurons very poor detectors, as plotted in Fig. 9 (a-b) for two of them.

We reproduce the reconstruction process at the population level and we plot in Fig. 9 c) the Euclidean error based on the number of neurons used during reconstruction and in Fig. 9 d) the results for different ensembles of neurons.

We can observe from the graph that near forty neurons are necessary in our case to recover the full resolution of the image without error. This number is, however, a large upper bound because according to the equation defined in the Supplementary Information: $NL \log L \geq L \log R$, only eight neurons need to have orthogonal codes as $N = \log 256 / \log 2 \approx 8$, which is in line with binary codes in bits in image processing.

The neural codes transform the pixel distribution into an independent identical distribution. Therefore, the spatial redundancy is removed, which makes the representation sparse.

Discussion

Biological plausibility. We described in this paper how artificial neural networks with discrete and ordinal synaptic weights can reconstrcut back missing information in original signals. However, we did not provide any grounds how the brain may implement this function. We suggest two plausible mechanisms.

First, chemical and electrical synapses possess specific strength and plasticity to conduct nerve impulses, and therefore particular synaptic resolution and potentiation. Above this potentiation limit, synapses may saturate, as it can be the case for binary synapses. This may induce a strong discretization of the incoming signal, which may be well rendered by discrete or low resolution weights as we suggest.

Second, while in many domains of human cognition, sequentially and hierarchically structured representations are thought to play a key role, many evidences suggest that the neurons in the frontal cortex are involved in their encoding (15, 16, 18, 54–56).

We review several neurocomputational models for ordinal encoding in (48) and provide as well an original biological implementation of it based on STDP, called ordinal STDP.

In this ordinal version of STDP, pre- and post-synaptic neurons reinforce their links with respect on their relative index, relative distance or relative spatial location, but not on their timing as it is conventionally the case in STDP.

Post-synaptic neurons with higher (resp. lower) index than pre-synaptic ones will strengthen (resp. diminish) their synaptic links. By doing so, spiking neurons sensitive to specific order in sequences can be constructed in recurrent networks, without the need to encode the indexes in the synapses as it is currently done in simulations. This type of ordinal encoding will have for advantage to be very robust to intrinsic noise even when nodes are inserted or suppressed within the sequence.

Digital computation. We found the counterintuitive result that a very limited number of neurons with coarse resolution and random connections can accurately encode arbitrary signals of very high resolution, one hundred million times higher than the neurons' learning capabilities, and this ratio grows linearly.

Because of the neurons' ordinal representation of the sequence, items taken in a repertoire of dimension R can be represented by a neural code with values taken in a repertoire of dimension L only, making the computation cycles of encoding and decoding very fast. Surprisingly, the principal bottleneck is at the reconstruction stage, which is dependent on the parameter σ and the number of neurons N used in the calculation of the Gaussian functions. The larger σ is, the heavier the computation but also the more precise the reconstruction, as shown in the experiments. The number of units required remains nonetheless low and almost constant.

Considering our finding that very few neurons are necessary to retrieve back items taken in very large repertoires, this mechanism is very sparse and computationally efficient.

From this result, we found a mathematical relationship between the memory capacity of a neural population — which depends on its number of neurons N and the resolution of its neural codes $R_{network} = L$ — and the repertoire cardinality R_{input} of the message to be learned (equivalently its entropy or resolution). This parallels Shannon's so-called separation theorem, which states that an information source may be transmitted (or stored) with an arbitrarily small frequency of error as long as the source entropy per channel use does not exceed the channel (or storage) capacity. The related equations are provided in the *SI* section; see Eqs. 1 and 2. These equations can provide insights for a neural information theory and the design of novel *digital* neural networks, for which the neural network we propose is one of the first instances to our knowledge.

The ordinal codes in our neural architecture operate a discretization (quantization) of information. Additionally, the randomly permuted orders yield independent and identically distributed rank items to be coded (with maximum entropy). This max entropy holds under the model assumption that the random permutations are chosen uniformly for each sequence learned. Together, they form a joint source-channel code that allows disambiguation of sequences that map to the same rank order in the original alphabet. This is in line with the efficient coding principle of redundancy reduction and the whitening phenomenon, which is found in sparse coding (3, 8, 11, 12, 57).

Error-correcting codes are at the heart of the revolution of modern communication theory and practice. Interestingly, the pioneering work that first came within reach of the Shannon limit to error-free communication is the turbo-codes (37), which exploit the three mechanisms we similarly use here: shuffling of the input signal, coarse representation (by parity-check codes), and belief votes at the decoder.

Within the brain, since the task of correctly retrieving a particular group of neurons is incommensurate (58) considering the number of neural units (10^{10} cells) and the number of synaptic connections (10^{14} dendrites), the brain has to find an efficient solution to overcome its complexity for processing, protecting and retrieving information (3). One outstanding and provocative question is, therefore, does the brain effectively use a digital code to process neural information? Although not a computer, does the brain incidentally exploit the same principles found in current man-made telecommunication networks?

After all, if *'DNA is encoded digital information in the "Strong Sense"'* according to Richard Dawkins (59), the brain may also exploit some kind of digital processing for memory preservation, access and learning.

Perceptual binding. Pouget and colleagues present in (60) a Bayesian inference treatment of feature binding between two or more variables emphasizing the important role of the Gaussian distribution of the neurons' output: that is, neurons encode (Gaussian) probabilities, and as such, they are sensitive to specific values in high dimensions (i.e., their mean value), which are a compromise between multiple inputs but weighted by the inverse of their variance (their precision or uncertainty). Therefore, in comparison with perceptrons in formal neural networks, biological neurons convey two types of information instead of one: the output and the confidence level.

They use one example in which the estimation of the width of an object (μ_{width}) is performed by combining visual and tactile cues (X_1 and X_2); the estimated size of the object is the average between these two variables with mean and variance: $\mu_{width} = W_1X_1 + W_2X_2$. As such, combining cues increases (or decreases) the information and confidence level.

We can draw a parallel with our framework. In our examples, we can see each element of the sequence as if they were different variables, modalities or cues, with access to different sources of information. The more items that are added in the sequence, the more active the neuron, which corresponds to its confidence level. Alternatively, the weights matrix W can also be read in the other direction, vertically: the more neurons that are used in element estimation, the more precise the reconstruction through accumulation (61).

The Gaussian functions and equations in our model are used in the same manner as the Bayesian treatment of uncertainty during feature binding in (60), with variance as a sign of computation (62). In addition, the randomization introduced in our algorithm has the advantage of creating a normal distribution of the variables so that each variable is independent and separable from the others. Sparsity is, therefore, a feature as important as binding for "tractability" or disentanglement in perception (63).

Information routing and conscious gating. Our ideas may cast light on recent proposals that the PFC is the brain router that may manipulate neuronal variables and pointers for gating information and conscious access (64, 65) or others that the brain manipulates integrated and differentiated information codes (66).

Because retrieval is viewed as an optimization decoding process in our network, our framework may explain why the bandwidth limitation to memory access is all-or-none and why conscious access is constrained, time-limited, and sequential. Under this aspect, it is in line with the predictive processing and free-energy account for consciousness, *in which consciousness is simply the process of optimizing beliefs through inference* (67–70). Their results are also in accordance with current main theories of the brain that relate conscious processing explicitly with information theory to global ignition, long-distance broadcasting, cognitive cost and information integration (66, 69, 71–76).

Our results may provide some additional constraints on the types of neural coding and communication mechanisms necessary for distal neurons to dynamically control the synchronization of a coherent neural assembly, to fulfill gating and conscious processing. Moreover, the efficiency of the working memory can be evaluated quantitatively in terms of durability and access to stored information: (1) its robustness against catastrophic forgetting and (2) its rapidity in retrieving any pieces of information, even corrupted ones.

Digital Neural Networks. Some other neural architectures have been proposed recently to incorporate discreteness and digital capabilities. Instances have been proposed by Berrou and colleagues (77) focusing on the problem of memory capacity and organization while Graves and colleagues (78, 79) investigated their computer-like features.

Berrou's neural network borrows the technique from telecommunication networks, showing high memory capac-

ity and sparsity, but its use in real case problems and its computational efficacy in real time have not been investigated. Additionally, Graves' neural Turing machine and Differentiable Neural Computer (DNC) show the computational capabilities of conventional computers with the use of neuronal pointers and access to external memory. The feature of having an addressable memory with pairs of (key,value) permits DNC to buffer and manipulate variables and codes, useful for symbolic AI problems. However, the problem of how the organization of memory *and* information processing can be efficiently combined, what the brain does, still remains.

To make a parallel with these neural networks and also with the von Neuman's computer architecture, we may see the neural population we introduced to realize the function of a random access memory (RAM) to retrieve quickly the *addresses* of neurons in a large-scale memory network. These neural addresses are represented in our neural architecture by quantized ordinal codes. Our results showed that a relatively small neural code can retrieve sequences in a very large-scale repertoire; e.g., an external memory system.

Conclusion

Our proposal that the brain may manipulate and compute a kind of *digital* information, may remind the pioneering and provocative works of the founders of computers and computation, John von Neumann and Alan Turing, to which we can add Claude Shannon. On the one hand, John von Neumann (80) created the standard model of computer architectures based on the separation between the operative and the operand, with memory-stored control and memory-stored programs. He also suggested the idea that the brain might be necessarily a digital parallel addressable memory machine to avoid noise, to keep and compute information. On the other hand, Alan Turing was perfectly aware of the cost of computation that has to endure the human memory system (81), "necessarily limited" (82) (p.231), to process, retrieve and keep track of information. Besides, noise, storage and transmission are at the heart of the concerns of the Communication and Information Theory of Claude Shannon (38), who also worked on the redundancy of the English language (83), in Artificial Intelligence (chess and maze-solving programs) and information storage in genes before the discovery of DNA.

Before Claude Shannon's work, engineers thought that to reduce communications errors, it was necessary to increase transmission power or to send the same message repeatedly. Shannon basically showed it was not necessary to waste so much energy and time if you had the right coding schemes (38).

Current machine learning techniques (e.g. deep networks) rely extensively on big data and large neural networks to approximate statistical correlations on a relatively small number of classes (a few hundred). It is acknowledged that we will soon arrive at the end of a cycle as it becomes harder to achieve significant improvements with the difficulty of accessing a larger volume of data and constructing larger deep networks. Furthermore, energy consumption becomes problematic as powerful computers, graphic cards, and high-performance computing are now required for efficient learning in a reasonable amount of time with those models. Thus, we can make a parallel with the current situation in AI and the earlier ages before digital communication.

Current AI architectures (deep networks) mostly ignore that computating has physical means and energy costs that biological systems cannot afford as they do not have access to a virtually unlimited amount of energy, precision and time, and have an urge to act. According to the efficient coding hypothesis for the brain, leveraging maximum entropy may be a decisive ally to achieve neat computational power with limited resources (neurons and synapses). We suggest that this leap be done by the digitalization of information for energy consumption, computational effectiveness and preserving information.

Data Availability.

The code to run the neural network is available on GitHub (https://git.u-cergy.fr/neurocyber/digicode).

ACKNOWLEDGMENTS. IPAL (M.Q.) and Labex MME-DII (M.Q.&A.P.). We would like to thank the two anonymous reviewers for their careful reading of our manuscript, their valuable comments and constructive remarks.

1. Barlow H (1961) Possible principles underlying the transformation of sensory messages. *Rosenblith, W. (Ed.), Sensory Communication. MIT Press, Cambridge, MA.*
2. Baddeley R, Hancock PJ, Földiák P (2000) *Information theory and the brain.* (Cambridge University Press Cambridge).
3. Laughlin S, Sejnowski T (2003) Communication in neuronal networks. *Science (New York, N.Y.)* 301:1870–4.
4. Olshausen BA, Field DJ (2004) Sparse coding of sensory input. *Curr Opin Neurobiol* 14(6):481–487.
5. Rolls E, Treves A (2011) The neuronal encoding of information in the brain. *Progress in Neurobiology* 95:448–490.
6. McClelland JL, McNaughton BL, O'Reilly RC (1995) Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review* 102:419–457.
7. Olshausen BA, Field DJ (1996) Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381:607–609.
8. Barlow H (2001) Redundancy reduction revisited. *Network* 12(3):241–253.
9. Olshausen B, Lewicki M (2013) What natural scene statistics can tell us about cortical representation. *The New Visual Neurosciences. J. Werner, L.M. Chalupa, Eds. MIT Press.*
10. Laughlin S (1981) A simple coding procedure enhances a neuron's information capacity. *Zeitschrift für Naturforschung. Section C: Biosciences* 36(9-10):910–912.
11. van Hateren J (1992) A theory of maximizing sensory information. *Biological Cybernetics* 68(1):23–29.
12. Atick J, Redlich AN (1992) What does the retina know about natural scenes? *Neural Computation* 4:196–210.
13. Rolls E (2016) Pattern separation, completion, and categorisation in the hippocampus and neocortex. *Neurobiol. Learn. Mem.* 129:4–28.
14. Barone P, J.P. J (2018) Prefrontal cortex and spatial sequencing in macaque monkey. *Exp Brain Res* 78:447– 464.
15. Genovesio A, Tsujimoto S, Wise S (2009) Feature- and order-based timing representations in the frontal cortex. *Neuron* 63:254–266.
16. Bakhurin K, et al. (2017) Differential encoding of time by prefrontal and striatal network dynamics. *The Journal of Neuroscience* 37(4):854 – 870.
17. Carpenter A, Baud-Bovy G, Georgopoulos A, Pellizzer G (2018) Encoding of serial order in working memory: Neuronal activity in motor, premotor, and prefrontal cortex during a memory scanning task. *The Journal of Neuroscience* 38(21):4912–4933.
18. Guidalia G, Pisonia, A. Bolognini N, Papagnoa C (2019) Keeping order in the brain: The supramarginal gyrus and serial order in short-term memory. *Cortex* 119:89–99.
19. Rolls E, Mills P (2017) The generation of time in the hippocampal memory system. *Cell Rep.* 28:1649–1658.
20. Liu Y, Dolan RJ, Kurth-Nelson Z, Behrens TE (2019) Human replay spontaneously reorganizes experience. *Cell* 178(3):640–652.e14.
21. Botvinick M, Watanabe T (2007) From numerosity to ordinal rank a gain-field model of serial order representation in cortical working memory. *The Journal of Neuroscience* 27(32):8636–8642.
22. Rigotti M, et al. (2013) The importance of mixed selectivity in complex cognitive tasks. *Nature* 497(7451):585–590.
23. Fusi S, Miller E, Rigotti M (2016) Why neurons mix: high dimensionality for higher cognition. *Curr. Opin. Neurobiol.* 37:66–74.
24. Ninokura Y, Mushiake H, Tanji J (2004) Integration of temporal order and object information in the monkey lateral prefrontal cortex. *Journal of neurophysiology* 91:555–60.
25. Shima K, Isoda M, Mushiake H, Tanji J (2007) Categorization of behavioural sequences in the prefrontal cortex. *Nature* 445:315–318.
26. Dehaene S, Meyniel F, Wacongne C, Wang L, Pallier C (2015) The neural representation of sequences from transition probabilities to algebraic patterns and linguistic trees. *Neuron* 88:2–19.
27. Richards BA, Frankland PW (2013) The conjunctive trace. *Hippocampus* 23:207–212.
28. Parthasarathy A, et al. (2017) Mixed selectivity morphs population codes in prefrontal cortex. *Nat. Neurosci.* 20:1770–1779.

29. Friston K, et al. (2016) Active inference and learning. *Neuroscience & Biobehavioral Reviews* 68:862–879.
30. Annabi L, Pitti A, Quoy M (2021) Bidirectional interaction between visual and motor generative models using predictive coding and active inference. *Neural Networks* 143:638–656.
31. Rao R, Ballard D (1999) Predictive coding in the visual cortex a functional interpretation of some extra-classical receptive-field effects. *Nat Neurosci* 2:79–87.
32. Spratling M (2016) Predictive coding as a model of cognition. *Cognitive Processing* 17(3):279–305.
33. Annabi L, Pitti A, Quoy M (2022) Continual sequence modeling with predictive coding. *Frontiers in Neurorobotics* 16.
34. Friston K (2009) The free-energy principle a rough guide to the brain? *Trends in Cognitive Science* 4(7):293–301.
35. Friston K, Kiebel S (2009) Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences* 364:1211–21.
36. Pitti A, Gaussier P, Quoy M (2017) Iterative free-energy optimization for recurrent neural networks (inferno). *PLoS ONE* 12(3):e0173684.
37. Berrou C, Glavieux A, Thitimajshima P (1993) Near shannon limit error-correcting coding and decoding: Turbo-codes. 1 in *Proceedings of ICC '93 - IEEE International Conference on Communications*. Vol. 2, pp. 1064–1070 vol.2.
38. Shannon CE (1948) A mathematical theory of communication. *Bell Systems Technical Journal. Available at: https://archive.org/details/bellsystemtechni27amerrich/page/379* 27(3):379–423, 623–56.
39. Guizzo E (2004) Closing in on the perfect code [turbo codes]. *IEEE Spectrum* 41(3):36–42.
40. Bi G, Poo M (1998) Activity-induced synaptic modifications in hippocampal culture, dependence of spike timing, synaptic strength and cell type. *J. Neurscience* 18:10464–10472.
41. Song S, Miller K, Abbott L (2000) Competitive hebbian learning and through spike-timing-dependent synaptic plasticity. *Nature neuroscience* 3:919–926.
42. Thorpe S, Delorme A, Van Rullen R (2001) Spike-based strategies for rapid processing. *Neural Networks* 14:715–725.
43. Van Rullen R, Thorpe S (2002) Surfing a spike wave down the ventral stream. *Vision Research* 42:2593–2615.
44. Pitti A, Quoy M, Boucenna S, Lavandier C (2021) Brain-inspired model for early vocal learning and correspondence matching using free-energy optimization. *PLOS Computational Biology* 17(2):1–27.
45. Abbott L, Nelson S (2000) Synaptic plasticity taming the beast. *Nature neuroscience* 3:1178–1182.
46. Izhikevich EM, Gally A. J, Edelman, G. M (2004) Spike-timing dynamics of neuronal groups. *Cerebral Cortex* 14:933–944.
47. Pitti A, Quoy M, Lavandier C, Boucenna S (2020) Gated spiking neural network using iterative free-energy optimization and rank-order coding for structure learning in memory sequences (inferno gate). *Neural Networks* 121:242–258.
48. Pitti A, et al. (2022) In search of a neural model for serial order: a brain theory for memory development and higher-level cognition. *IEEE Transactions on Cognitive and Developmental Systems* 10.1109/TCDS.2022.3168046.
49. Kohonen T (1982) Self-organized formation of topologically correct feature maps. *Biological Cybernetics* 43:59–69.
50. Enel P, Procyk E, Quilodran R, Dominey P (2016) Reservoir computing properties of neural dynamics in prefrontal cortex. *PLoS Comput Biol* 6:e1004967.
51. Nakajima K, Fischer I (2021) *Reservoir Computing*. (Springer, Eds. Singapore).
52. Dasgupta S, Stevens C, Navlakha S (2017) A neural algorithm for a fundamental computing problem. *Science* 358(6364):793–796.
53. French RM (1999) Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences* 3(4):128–135.
54. Petrides M (1995) Impairments on nonspatial self-ordered and externally ordered working memory tasks after lesions of the mid-dorsal part of the lateral frontal cortex in the monkey. *J Neurosci.* 15:359–75.
55. Fitch W, Martins M (2014) Hierarchical processing in music, language, and action: Lashley revisited. *Ann N Y Acad Sci* 1316(1):87–104.
56. Jeon HA (2014) Hierarchical processing in the prefrontal cortex in a variety of cognitive domains. *Frontiers in Systems Neuroscience* 8.
57. Olshausen BA, Field DJ (1997) Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research* 37:3311–3325.
58. Herculano-Houzel S (2009) The human brain in numbers: a linearly scaled-up primate brain. *Frontiers in Human Neuroscience* 3:31.
59. Dawkins R (1995) *River Out of Eden*.
60. Pouget A, Beck J, Ma W, Latham P (2013) Probabilistic brains knowns and unknowns. *Nature Neuroscience* 16(7):1170–1178.
61. Beck J, et al. (2008) Probabilistic population codes for bayesian decision making. *Neuron* 60:1142–1152.
62. Churchland A, et al. (2011) Variance as a signature of neural computations during decision making. *Neuron* 69(4):818–831.
63. Averbeck B, Latham P, Pouget A (2006) Neural correlations, population coding and computation. *Nat Rev Neurosci.* 7(5):358–66.
64. Zylberberg A, Slezak F, Roelfsema, P.R. Dehaene S, Sigman M (2010) The brain's router: A cortical network model of serial processing in the primate brain. *PLoS Comput Biol* 6(4):e1000765.
65. Dehaene S, Changeux J (2011) Experimental and theoretical approaches to conscious processing. *Neuron* 70:200–227.
66. Balduzzi D, Tononi G (2009) Qualia: The geometry of integrated information. *PLoS Comput Biol* 5(8):e1000462.
67. Hobson JA, Friston KJ (2016) A response to our theatre critics. *Journal of Consciousness Studies* 23(3-4):245–254.
68. Clark A, Friston K, Wilkinson S (2019) Bayesing qualia: consciousness as inference, not raw datum. *Journal of Consciousness Studies* 26(9-10):19–33.
69. Kanai R, et al. (2019) Information generation as a functional basis of consciousness. *Neuroscience of Consciousness* 5(1):niz16.
70. Whyte CJ, Smith R (2020) The predictive global neuronal workspace: A formal active inference model of visual consciousness. *bioRxiv*.
71. Dehaene S, Kerszberg M, Changeux J (1998) A neuronal model of a global workspace in effortful cognitive tasks. *Proceedings of the national Academy of Sciences* 95(22):14529–14534.
72. Baars B (2005) Global workspace theory of consciousness: toward a cognitive neuroscience of human experience. *Progress in brain research* 150:45–53.
73. Tononi G (2008) Consciousness as integrated information: a provisional manifesto. *Biol. Bull.* 215:216–242.
74. Toker D, Sommer F (2019) Information integration in large brain networks. *PLoS Comput Biol* 15(2):e1006807.
75. Koechlin E, Summerfield C (2007) An information theoretical approach to prefrontal executive function. *Trends in Cognitive Sciences* 11(6):229–235.
76. Zénon A, Solopchuk O, Pezzulo G (2019) An information-theoretic perspective on the costs of cognition. *Neuropsychologia* 123:5–18. Cognitive Effort.
77. Berrou C, Dufor O, Gripon V, Jiang X (2014) Information, noise, coding, modulation: What about the brain? *8th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)* 44(6):167–172.
78. Graves A, Wayne G, Danihelka I (2014) Neural Turing Machines. *arXiv* 1410.541v2:1–26.
79. Graves Aea (2016) Hybrid Computing Using a Neural Network with Dynamic External Memory. *Nature* 538:471–476.
80. von Neumann J (1958) The computer and the brain. *New Haven, CT: Yale University Press*.
81. Turing A (1950) letter in response to a. m. uttley's talk on 'information, machines and brains', conference on information theory, 26-29 september 1950.
82. Turing A (1936) On computable numbers, with an application to the entscheidungsproblem. *Proc. Lond. Math. Soc.* 2:230–265.
83. Shannon CE (1951) Prediction and entropy of printed english. *Bell System Technical Journal* 30(1):50–64.