Supplementary Note

Perceptron Learning and Capacity.

The perceptron\textsuperscript{1–3} is a node that makes a binary decision by thresholding a weighted sum of its \(N\) inputs. For convenience, in this Supplementary Note we denote the decision variables by \(\sigma \in \{0, 1\}\) and not by \(\oplus\) and \(\ominus\) as in the paper. The inputs are given by a vector \(\vec{x} = (x_1, \ldots, x_N)\), where each component is a single scalar that can be interpreted as the firing rate of an afferent neuron. Thus, the decision of the perceptron is given by

\[
\sigma = \Theta \left( \sum_{i=1}^{N} \omega_i x_i - \vartheta \right),
\]

where \(\Theta\) is the Heaviside step function, \(\omega_i\) denotes the weight of the \(i\)-th input component and \(\vartheta\) is the decision threshold.

A labeled pattern consists of an input vector \(\vec{x}\) and a scalar \(y \in \{0, 1\}\) that denotes the desired output for this input. Given a set of \(p\) labeled patterns \(\{(\vec{x}^\mu, y^\mu) | \mu = 1, \ldots, p\}\), the perceptron learning rule consists of the following synaptic changes: For each input pattern \(\vec{x}^\mu\) for which the perceptron decision \(\sigma^\mu\) is not equal to the desired output \(y^\mu\), change all weights \(\omega_i (i = 1, \ldots, N)\) according to

\[
\omega_i \mapsto \omega_i + \lambda x_i^\mu (2y^\mu - 1).
\]

If a vector of weights that correctly classifies all patterns exists, then this rule converges to some solution vector after a finite number of cycles through all \(p\) labeled patterns. It is well-known that this rule converges in a finite number of cycles through all the \(p\) labeled patterns to a vector of weights that correctly classify all of them provided that such a vector exists. For large \(N\) and random patterns (see below) such a solution exists with probability 1 if and only if \(\alpha = p/N \leq \alpha_c\), where \(\alpha_c = 2\) (ref.\textsuperscript{4}).

The perceptron capacity of \(\alpha_c = 2\) holds for patterns that are randomly labeled \(y = 0\) or \(y = 1\) with equal probability, but is quite insensitive to the statistics of the inputs \(x_i\). To show this, we consider the case of binary inputs that assume values \(\xi_i = 0\) and \(\xi_i = 1\) with respective probabilities \((1 + \text{m}_{\text{in}})/2\) and \((1 - \text{m}_{\text{in}})/2\). Each pattern is labeled as \(y = 0\) and \(y = 1\) with respective probabilities \((1 + \text{m}_{\text{out}})/2\) and \((1 - \text{m}_{\text{out}})/2\). Following the derivation of the perceptron capacity in the limit of large \(N\) in ref.\textsuperscript{4}, we obtain for the present case,(cf. Eq. 34-40 in ref.\textsuperscript{4})

\[
\frac{1}{\alpha_c} = \frac{1}{2} (1 + \text{m}_{\text{out}}) \int_{-x}^{x} D(t-x)^2 + \frac{1}{2} (1 - \text{m}_{\text{out}}) \int_{-x}^{x} D(t+x)^2.
\]

Here \(x\) is an order parameter which is determined through the following saddle-point equation,

\[
\frac{1}{2} (1 + \text{m}_{\text{out}}) \int_{x}^{\infty} D(t-x) = \frac{1}{2} (1 - \text{m}_{\text{out}}) \int_{-x}^{\infty} D(t+x).
\]
This equation determines the value of $x$ and hence of $\alpha_c$ as functions of $m_{\text{out}}$ alone, independently of $m_{\text{in}}$. In particular, for $m_{\text{out}} = 1/2$, the solution of the above equation is $x = 0$, which upon substitution in Eq. (3) yields $\alpha_c = 2$ for all $0 < m_{\text{in}} < 1$. Thus, $\alpha_c = 2$ is a useful benchmark for the capacity of learning random patterns with unbiased labels, such as considered in this paper.

**Discrete tempotron**

A discrete time version of the tempotron (Supplementary Fig. 1), which is more amenable to statistical mechanical analysis can be devised by segmenting the input time interval $[0, T]$ into $k$ disjoint time slices $[\lambda_l, \lambda_{l+1}]$ with $l = 1, \ldots, k$ and boundaries $\lambda_1 = 0$ and $\lambda_{k+1} = T$. The number of spikes that arrive at afferent $i$ during a given time segment $l$ is denoted by $\xi^l_i$. For each $l$ the $N$ variables $\xi^l_i$ serve as inputs into a standard perceptron whose output $\zeta^l_i$ is (cf. Eq. 1)

$$\zeta^l_i = \Theta(h^l_i - \vartheta), \quad (5)$$

where the field

$$h^l_i = \sum_{i=1}^{N} \omega_i \xi^l_i. \quad (6)$$

Note that all $k$ perceptrons in this layer share the same set of weights $\omega_i$. A single second layer perceptron computes the final decision $\sigma$ by thresholding the sum of all $\zeta^l_i$,

$$\sigma = \Theta\left(\sum_{l=1}^{k} \zeta^l_i - \frac{1}{2}\right). \quad (7)$$

Hence, the discrete tempotron gives $\sigma = 1$ if at least one perceptron in the first layer results in $\zeta^l_i = 1$ and 0 if all first layer perceptrons give 0.

In the discrete tempotron learning is confined to the first layer. By analogy with the continuous tempotron, which aligns synaptic changes to the post-synaptic voltage maximum, the discrete tempotron learning algorithm targets those weights that contribute to the maximal field in the first layer. Specifically, on each error trial the first-layer perceptron $l_{\text{max}}$ where

$$l_{\text{max}} = \arg \max_l h^l_t,$$

is subjected to a standard perceptron weight update (cf. Eq. 2) with the desired tempotron output taking the role of the desired single perceptron output.