

Lecture 22

Capacity of the Hopfield network

(HN)

I neurons, N memories

Consider a binary HN, with the Hebbian learning rule.

Let's say the network is set to pattern $\vec{x}^{(n)}$ which was generated randomly: (memory)

$$a_i = \sum_{j \neq i} w_{ij} x_j^{(n)},$$

$$w_{ij} = \underbrace{x_i^{(n)} x_j^{(n)}}_{\substack{\eta=1 \\ \text{here}}} + \underbrace{\sum_{m \neq n} x_i^{(m)} x_j^{(m)}}_{\substack{\text{"noise"} \\ \text{(reinforces } \vec{x}^{(n)})}}$$

Then
$$a_i = \sum_{j \neq i} x_i^{(n)} \underbrace{x_j^{(n)} x_j^{(n)}}_{=1} +$$

$$+ \sum_{j \neq i} \sum_{m \neq n} x_i^{(m)} x_j^{(m)} x_j^{(n)} = \underbrace{(I-1)}_{\substack{\text{desired state } x \\ I-1 \Rightarrow \text{keeps node } i \\ \text{clamped to} \\ \text{the correct} \\ \text{value } x_i^{(n)}}} x_i^{(n)} +$$

$$+ \sum_{j \neq i} \sum_{m \neq n} x_i^{(m)} x_j^{(m)} x_j^{(n)}$$

sum of $(I-1)(N-1)$ random variables (by assumption) independent

Now, we have, by construction,

$$x_k^{(p)} = \begin{cases} +1, & p = \frac{1}{2} \\ -1, & p = \frac{1}{2} \end{cases} \\ \forall k, \forall p$$

Then $\underbrace{x_i^{(m)} x_j^{(m)} x_j^{(n)}}_{\text{all 3 terms distinct}} \\ (i \neq j, m \neq n)$

8 configurations:

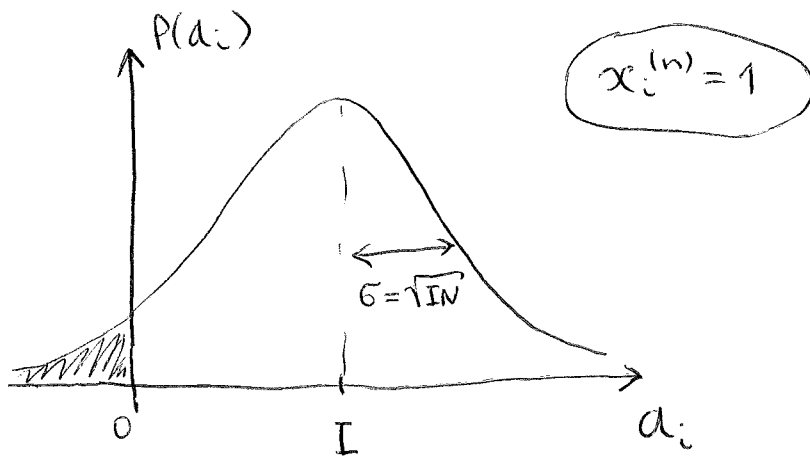
$$\left\{ \begin{array}{l} 1 \ 1 \ 1 \\ 1 \ 1 \ -1 \\ 1 \ -1 \ 1 \\ 1 \ -1 \ -1 \\ -1 \ 1 \ 1 \\ -1 \ 1 \ -1 \\ -1 \ -1 \ 1 \\ -1 \ -1 \ -1 \end{array} \right. \quad p = \frac{1}{8} \text{ each} \Rightarrow \left\{ \begin{array}{l} 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \\ -1 \end{array} \right. \xrightarrow{\text{product}} \begin{cases} +1, p = \frac{1}{2} \\ -1, p = \frac{1}{2} \end{cases}$$

$$\text{So, } \underbrace{\langle x_i^{(m)} x_j^{(m)} x_j^{(n)} \rangle}_{\text{mean}} = 0$$

$$\text{Var} \{ x_i^{(m)} x_j^{(m)} x_j^{(n)} \} = \sum_{i=1}^8 \frac{1}{8} 1 = 1$$

Central limit theorem:

$$P(d_i) = \mathcal{N}(d_i \mid \underbrace{(I-1)x_i^{(n)}}_{\mu}, \underbrace{(I-1)(N-1)}_{\sigma^2}) \approx \\ \approx \mathcal{N}(d_i \mid I x_i^{(n)}, I N)$$



$$P(i \text{ is unstable}) = P(a_i < 0) \quad \textcircled{=}$$

Spin will flip
in a single iteration

$$\frac{1}{\sqrt{IN}} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^0 da_i e^{-\frac{(a_i - I)^2}{2(IN)}} = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{-\frac{I}{\sqrt{IN}}} dz \sqrt{IN} e^{-\frac{z^2}{2}}$$

$$z = \frac{a_i - I}{\sqrt{IN}}$$

$$\textcircled{=} \Phi\left(-\frac{I}{\sqrt{IN}}\right) = \Phi\left(-\frac{1}{\sqrt{N/I}}\right)$$

Usually, $\frac{N}{I} \ll 1 \Rightarrow \frac{1}{\sqrt{N/I}} \gg 1$.

$$\left[\text{Use } \Phi(-z) \approx \frac{1}{\sqrt{2\pi}} \frac{e^{-z^2/2}}{z}, \quad z \gg 1 \right] \quad (*)$$

Then, if we require the total error to be $\frac{\epsilon}{b}$ (all memories, all spins):

$$\Phi\left(-\frac{I}{\sqrt{NI}}\right) \leq \frac{\epsilon}{NI}$$

error per spin flip

Using (*), we obtain:

$$[z = \sqrt{\frac{I}{N}}]$$

$$\frac{1}{\sqrt{2\pi}} \sqrt{\frac{N}{I}} e^{-\frac{1}{2} \frac{I}{N}} \lesssim \frac{z}{NI}, \text{ or}$$

$$\log \frac{1}{\sqrt{2\pi}} + \frac{1}{2} (\log N - \log I) - \frac{1}{2} \frac{I}{N} \lesssim \log z - \log N - \log I,$$

$\ll \log I$
 $\ll \log I$

$$+ \frac{1}{2} \log I - \log z \lesssim \frac{1}{2} \frac{I}{N},$$

$$\frac{I}{N} \gtrsim \log I - 2 \log z, \text{ or}$$

$$\left[N \lesssim \frac{I}{\underbrace{\log I - 2 \log z}_{N_{\max}}} \right].$$

This analysis focuses on single-bit flips \Rightarrow no info about subsequent iterations (avalanches?)

Amit et al. PRL, 1985

spin-glass analysis of HN in the large I limit

$\frac{N}{I} > 0.138$ stable states ^{exist but are} uncorrelated with desired memories

$\frac{N}{I} \in (0, 0.138)$ there are stable states "close" to the desired memories

\mathcal{H}_0 , in addition, $\frac{N}{I} \in (0, 0.05)$ desired states are lower in energy than spurious states. free

Correspondingly, if

$\frac{N}{I} \in (0.05, 0.138)$ spurious states dominate (some of them have lower free energies than desired states)

$\frac{N}{I} \in (0, 0.03)$ mixture states appear (combinations of several desired states) but they are higher in free energy than the desired states

Can we do better than the Hebbian learning rule?

Make an objective function that, for every pattern $\vec{x}^{(n)}$, if $x_j = x_j^{(n)}$ for $\forall j \neq i$ (all ^{i.e.} neurons other than i are set correctly) \Rightarrow we prefer (i.e. assign higher score to) $x_i = x_i^{(n)}$.

So, try

$$E(W) = - \sum_{i=1}^I \sum_{n=1}^N \left[t_i^{(n)} \log y_i^{(n)} + (1 - t_i^{(n)}) \log (1 - y_i^{(n)}) \right], \text{ where}$$
$$t_i^{(n)} = \begin{cases} 1 & \text{if } x_i^{(n)} = 1, \\ 0 & \text{if } x_i^{(n)} = -1 \end{cases}$$

Furthermore,

$$y_i^{(n)} = \frac{1}{1 + e^{-a_i^{(n)}}}, \quad a_i^{(n)} = \sum_{j \neq i} w_{ij} x_j^{(n)}$$

Just like classification of each bit in each memory into +1/-1 classes ($K=2$). (!) We can use logistic regression to fit the weights & use those ~~weights~~ weights to build the NN.

[Stores more memories (empirically)]
[than the Hebbian rule.]

Boltzmann machines (BM)

Consider

$$\begin{cases} E(\vec{x}) = -\frac{1}{2} \sum_{i,j} x_i w_{ij} x_j = -\frac{1}{2} \vec{x}^T W \vec{x} \\ P(\vec{x}) = \frac{1}{Z} e^{-E(\vec{x})} \end{cases}$$

Stochastic Hopfield network (aka Boltzmann machine, BM) ← actually implements Boltzmann distr'n

Activity rule:

$$a_i = \sum_j w_{ij} x_j$$

$$x_i = \begin{cases} +1, & \text{prob. } q_i = \frac{1}{1+e^{-2a_i}} = \frac{e^{a_i}}{e^{a_i}+e^{-a_i}} \\ -1, & \text{prob. } 1-q_i = \frac{e^{-a_i}}{e^{a_i}+e^{-a_i}} \end{cases}$$

→ stochastic update

Gibbs sampling

cf. p. 402 31.1

Consider

$$E(\vec{x}) = -\frac{1}{2} J \sum_{\substack{m,n \\ m \neq n}} x_m x_n - H \sum_n x_n$$

→ spin glass

Then $b_n = J \sum_{\substack{m \\ m \neq n}} x_m + H$ is the local field for spin n .

Indeed, for 2 spins

$$\begin{aligned} E &= -\frac{J}{2} (x_1 x_2 + x_2 x_1) - H(x_1 + x_2) = \\ &= -x_2 (Jx_1 + H) - Hx_1 = \\ &= -x_1 (Jx_2 + H) - Hx_2 \end{aligned}$$

In general,
 $E = -x_n b_n + \text{const}(x_n)$

Gibbs sampling: select spin n at random

$$P(S_n = +1 | b_n) = \frac{e^{+\beta b_n}}{e^{+\beta b_n} + e^{-\beta b_n}} = \frac{1}{1 + e^{-2\beta b_n}}$$

↑
all other spins
fixed

$$P(S_n = -1 | b_n) = 1 - P(S_n = +1 | b_n)$$

Use these probabilities to set the spin state: ± 1 .

This converges to Boltzmann equilibrium.

Metropolis sampling:

Compute $\Delta E = \begin{cases} x_n = 1 \Rightarrow x_n = -1 : E = -b_n + \text{const} \Rightarrow b_n + \text{const} : \Delta E = -2b_n \\ x_n = -1 \Rightarrow x_n = 1 : E = b_n + \text{const} \Rightarrow -b_n + \text{const} : \Delta E = 2b_n \end{cases}$

So, $\Delta E = 2b_n x_n$.

$P(\text{accept spin flip}) = \begin{cases} 1 & \Delta E \leq 0 \\ e^{-\beta \Delta E} & \Delta E > 0 \end{cases}$
This converges to Boltzmann eq'm as well.

Now, given a set of N examples $\{\vec{x}^{(n)}\}_{n=1}^N$, we might adjust weights w s.t. the likelihood of generating those examples from the Boltzmann distribution $P(\vec{x})$ is maximized:

$$\mathcal{Z} = \prod_{n=1}^N P(\vec{x}^{(n)}) \quad , \quad \text{or}$$

$$\log \mathcal{Z} = \sum_{n=1}^N \log P(\vec{x}^{(n)}) = \sum_n \left[\frac{1}{2} \vec{x}^{(n)T} W \vec{x}^{(n)} - \log z \right].$$

We need

$$\begin{aligned} \frac{\partial}{\partial w_{ij}} \log z &= \frac{1}{z} \frac{\partial}{\partial w_{ij}} \left\{ \sum_{\vec{x}} e^{-\beta E(\vec{x})} \right\} = \\ &= - \sum_{\vec{x}} P(\vec{x}) \frac{\partial}{\partial w_{ij}} E(\vec{x}) = \sum_{\vec{x}} x_i x_j P(\vec{x}) = \\ &= \langle x_i x_j \rangle_P. \end{aligned}$$

Then

$$\begin{aligned} \frac{\partial}{\partial w_{ij}} \log \mathcal{Z} &= \underbrace{\sum_n x_i^{(n)} x_j^{(n)}}_{N \langle x_i x_j \rangle_D} - N \langle x_i x_j \rangle_P = \\ &= N \left[\underbrace{\langle x_i x_j \rangle_D}_{\text{empirical 2-point correl'n}} - \underbrace{\langle x_i x_j \rangle_P}_{\text{model 2-point correl'n}} \right]. \end{aligned}$$

$$\text{If } \frac{\partial}{\partial w_{ij}} \log \mathcal{Z} = 0 \Rightarrow \langle x_i x_j \rangle_D = \langle x_i x_j \rangle_P.$$

↑ compute directly
↑ estimate by gibbs sampling

otherwise,

$\langle x_i x_j \rangle_D - \langle x_i x_j \rangle_P$ provides the gradient for optimization algorithms.

Note that if $W=0 \Rightarrow E(\bar{x})=0, \forall \bar{x}$.

Then

$$\langle x_i x_j \rangle_P = \langle x_i \rangle_P \langle x_j \rangle_P = 0,$$

since all spins are equally likely to be up or down.

↑
like the $\beta=0$
limit
(infinite T)

If the weights are adjusted by the gradient descent,

$$w_{ij}^{(\tau+1)} = w_{ij}^{(\tau)} + \eta \frac{\partial}{\partial w_{ij}} \log Z \Big|_{w_{ij}^{(\tau)}}$$

↑
learning rate, >0
guaranteed to increase $\log Z$ if

the step is small:

$$\log Z(w_{ij}^{(\tau+1)}) \approx \log Z(w_{ij}^{(\tau)}) +$$

$$+ \eta \underbrace{\frac{\partial}{\partial w_{ij}} \log Z \Big|_{w_{ij}^{(\tau)}} \times \frac{\partial}{\partial w_{ij}} \log Z \Big|_{w_{ij}^{(\tau)}}}_{\geq 0}$$

≥ 0 if $\eta > 0$

Thus in the $W=0$ case,

$$w_{ij}^{(1)} = w_{ij}^{(0)} + \eta \sum_n x_i^{(n)} x_j^{(n)}$$

← $w_{ij}^{(0)} = 0$, say

Hebbian learning rule is recovered in 1 iteration