Properties of associative memories

Classical Hopfield model with Hebb rule:

• the efficiency of the memory (drastically) degrades along with the increase of the number of stored patterns

• the efficiency degrades also in case of storing rare/dense patterns

➔Alternative learning rules have been proposed with the architecture of the network and its operational principles (in the restoring mode) remaining unchanged

Modifications of associative memory

(i) Matrix T is generated under the scheme other than the Hebb rule, and each pattern is presented only once.

(ii) Matrix T is generated iteratively – each pattern is presented multiple times.

(iii) T is generated under one of the classical rules (e.g. Hebb) but with individual choice of threshold for each neuron.

(iv) T is defined based on modified (preprocesses) set of patterns - optimised for the efficacy of the storing/restoring processes.

→ Capacity increases, however usually with the cost of loosing locality of the method and its biological interpretation.

(i) Single presentation of patterns

Pseudoinverse rule

$$T = XX^{+}, \qquad X^{+} = \lim_{\delta \to 0} (X^{T} X + \delta^{2} I)^{-1} X^{T}$$

X – matrix

In case X^i are linearly independent:

$$T = X(X^{T} X)^{-1} X^{T}$$ (the so called projection rule)

The rule is non-local, and offline.

Basic disadvantage: (?)

Projection rule – iterative version

$$T^s = T^{s-1} + \frac{1}{(X^s)^T X^s - [(X^s)^T T^{s-1} X^s]} [T^{s-1} X^s - X^s][T^{s-1} X^s - X^s]^T$$

$$T^0 = 0$$

Single presentation of patterns, the rule is ... online, non-local.

The capacity od the memory with the pseudoinverse rule – for linearly independent patterns = N

The attraction radius = $\dfrac{N}{2M}$

➔ For $M \approx \dfrac{N}{2}$ the memory loses completely correction property

Some improvement is possible when the diagonal of the matrix T is multiplied by a coefficient D from the interval (0.1, 0.2).

Avoiding the inverse matrix calculation

If X is a set of linearly independent vectors:

$$T = X(X^T X)^{-1} X^T \approx T_1 + T_2$$

where $T_1 = XX^T, \qquad T_2 = -\frac{1}{N} XBX^T$

B is defined as follows: $b_{ij} = (1-\delta_{ij})(X^i)^T X^j$

If $A = X^T X$ then $A = N[I + \frac{B}{N}]$

For $\frac{b_{ij}}{N} << 1$ we have $A^{-1} \approx \frac{1}{N}[I - \frac{B}{N}]$

Avoiding the inverse matrix calculation

Omitting scalar $\frac{1}{N}$ one obtains $(X^T X)^{-1} \approx [I - \frac{B}{N}]$

i.e. $$T \approx XX^T - \frac{1}{N} XBX^T$$

The rule is non-local and offline.

Hebb rule (T_1) with correction (T_2).

Much more efficient for uniformly distributed vectors compared to Hebb rule. Some improvement also in case or rare patterns.

Mutual inhibition method

Bipolar patterns. A special „inhibition" term is added to the Hebb rule:

$$t_{ij} = \begin{cases} \sum_{s=1}^{M} x_i^s x_j^s - \lambda \sum_{s=1}^{M} \sum_{p \neq s}^{M} x_i^p x_j^s & \quad for \quad i \neq j \\ 0 & \quad for \quad i = j, \end{cases} \qquad \lambda > 0.$$

Interpretation:

Rule local, offline.

Applied in case of ...
rare patterns – capacity is approximately doubled.

Modifications of associative memory

(i) Matrix T is generated under the scheme other than the Hebb rule, and each pattern is presented only once.

(ii) Matrix T is generated iteratively – each pattern is presented multiple times.

(iii) T is generated under one of the classical rules (e.g. Hebb) but with individual choice of threshold for each neuron.

(iv) T is defined based on modified (preprocesses) set of patterns - optimised for the efficacy of the storing/restoring processes.

(ii) Iterative methods

The matrix T is repeatedly improved by iterative (cyclic) Presentation of all training patterns.

Stopping condition: $\| T^M(t) - T^M(t-1) \| < \varepsilon$ *or* $t > t_{\max}$

$\| T^i(t) \|, i = 1, \ldots, M$ is the norm of matrix T after presentation of the ***i***-th training pattern at iteration ***t***.

$t_{\max}$ is the limit for the number of iterations.

Δ rule

Efficient variant of pseudoinverse rule.

$$T^k(t) = T^{k-1}(t) + \frac{\eta}{N}\left[X^k - T^{k-1}(t)X^k\right]\left[X^k\right]^T, \qquad t = 1,2,\ldots, \quad t < t_{\max}$$

$$T^0(1) = 0, \qquad \eta \in [0.7, 0.9]$$

Interpretation:

Excellent recognition properties. Low sensitivity in case of noisy input images. Efficiency comparable with the „original" pseudoinverse rule!

Minus – multiple presentation of all patterns is required.

Modified perceptron rule - MPR

Based on perceptron learning rule + weights symmetry.

Suppose no symmetry in weights is required, then T can be defined so as to fulfil:

$$X^k = \mathrm{sgn}(TX^k), \quad k = 1, \ldots, M$$

➔The problem of training *perceptron* with ***N*** inputs and ***N*** outputs to realize the *identity function* on ***X***.

Modified perceptron rule - MPR

„Classical" perceptron rule:

➔patterns are presented one by one and after presentation of patetrn $\boldsymbol{X}^{\mathbf{k}}$ the weights are being changed in the following way:

$$t_{ij}(t+1) = t_{ij}(t) + \Delta t_{ij}(t) = t_{ij}(t) + \eta(x_i^k - y_i^k)x_j^k \qquad for \quad i,j = 1,\dots,N$$

➔ Introduction of symmetry: $t_{ij} = t_{ji}$: $\frac{1}{2}(\Delta t_{ij}(t) + \Delta t_{ji}(t))$

$$\boxed{t_{ij}(t+1) = t_{ji}(t+1) = t_{ij}(t) + \frac{\eta}{2}\left[(x_i^k - y_i^k)x_j^k + (x_j^k - y_j^k)x_i^k\right]}$$

Modified perceptron rule - MPR

Proof of convergence analogous to the proof for perceptron rule.

Common features with the Hebb rule:

- locality
- biological soundness (connections between ... are strenghten)
- implementation feasibility

The main difference:

MPR is a supervised learning rule. MPR generalizes unsupervised Hebb rule by adding the term responsible for on-line error correction in the learning process.

Efficiency of MPR

1. Stability of base vectors.

2. Error correction.

<u>Automatic generation of base (training) vectors:</u> $< B, p >$

B - block p – probability of a 1's block

$< B, p >=< 1, 0.5 >$ - uniform patterns

$< B, p >=< 1, 0.1 >$ - rare patterns

$< B, p >=< 5, p >$ - correlated patterns

Stability of base vectors

Absolute (100%) stability for M=20, 40, 60 vectors of size N=200

M	B	p=0.1		p=0.3		p=0.5	
		bit	iter.	bit	iter.	bit	iter.
20	1	20.25	5.6	29.46	3.4	0.04	3.2
	3	20.73	5.9	21.75	3.4	1.36	3.2
	5	20.32	5.5	19.00	3.8	3.62	3.4
40	1	20.41	7.1	50.72	4.6	1.24	3.5
	3	20.49	6.5	40.53	4.9	4.30	4.6
	5	20.55	5.5	32.61	4.6	5.18	4.5
60	1	20.06	8.4	57.30	5.4	3.37	4.8
	3	20.22	7.9	48.54	6.2	5.37	6.0
	5	20.52	5.9	38.00	5.0	4.82	5.5

Average among 10 test sets

iter. – the average number of iterations in MPR necessary to achieve stable state.

bit – the average number of uncorrect bits under the Hebb rule.

Stability of base vectors

Stability of base vectors in MPR is much better than in the Hebb rule.

$$For \quad <B,p>=<1,0.5> \quad with \quad M \approx N\sqrt{N}$$

MPR leads to diagonally dominant matrix, i.e. ...

When the i-th bit in the training pattern (e.g. $X^{\mathbf{k}}$) is not stable, i.e.

$$x_i^k \neq y_i^k \qquad \text{then} \qquad t_{ii}(t+1) = t_{ii}(t) + 2\eta$$

Since the vectors are presented multiple times, when assuming long enough training time one obtains a diagonally dominant matrix:

$$|t_{ii}| > \sum_{j=1,\, j\neq i}^{N} |t_{ij}|, \qquad i = 1,\ldots,N$$

Error correction (ZRP=MPR)

Very good correction properties up to 25% disturbed (noisy) bits.

M	B	p=0.1		p=0.3		p=0.5	
		Hebb	ZRP	Hebb	ZRP	Hebb	ZRP
20	1	20.25	0.13	29.33	0.14	0.05	0.11
	3	20.73	0.13	22.61	0.15	1.47	0.13
	5	20.32	0.10	18.75	0.05	3.92	0.12
40	1	20.41	0.27	50.54	0.29	1.36	0.17
	3	20.49	0.15	40.38	0.27	4.34	0.27
	5	20.55	0.13	32.32	0.32	5.12	0.23
60	1	20.06	0.33	57.22	0.41	3.64	0.39
	3	20.22	0.23	48.35	0.41	5.47	0.36
	5	20.52	0.21	37.82	0.21	4.80	0.37

The number of erroneous bits after one-step synchronous recognition.

Results < 1 denote Partial correction.

Results > 1 denote the increase of the number of incorerct bits.

One randomly chosen bit is flipped in each pattern. Average over 10 sets.

Error correction (ZRP=MPR)

b	B	p=0.1		p=0.3		p=0.5	
		Hebb	ZRP	Hebb	ZRP	Hebb	ZRP
10	1	20.25	1.13	28.87	1.25	0.17	1.02
20		20.25	2.66	28.72	3.17	0.46	2.20
30		20.25	4.68	28.23	5.57	1.19	4.65
50		20.25	11.76	27.58	12.13	4.57	10.20
100		20.25	43.92	30.56	39.76	23.40	33.15
10	5	20.32	1.07	18.90	1.07	4.45	1.52
20		20.32	1.82	18.82	1.95	4.75	2.42
30		20.32	3.97	20.10	3.85	6.42	4.47
50		20.32	7.55	19.50	8.45	9.60	7.82
100		20.32	30.70	24.27	30.10	21.62	27.27

N=200

The number of errorneous bits after one-step synchronous recognition.

Results < b denote partial correction.

10 test sets.

b randomly chosen bits were flipped in each pattern.

MPR - summary

Rule is local, offline.

[1] Except for the case <B,p> = <1, 0.5> MPR visibly outperforms Hebb rule in error correction between 1 to 25% of bits.

[2] An important advantage of MPR is very low sensitivity to vectors' degree of density – on the contrary to the Hebb rule.

[3] For the Hebb rule, for p=0.1 approximately pN bits is unstable due to the convergence to a very strong attractor $\{-1\}^N$
➔ Hebb rule becomes useless. MPR remains effective in the range up to 25% of patterns' length.

Modifications of associative memory

(i) Matrix T is generated under the scheme other than the Hebb rule, and each pattern is presented only once.

(ii) Matrix T is generated iteratively – each pattern is presented multiple times.

(iii) T is generated under one of the classical rules (e.g. Hebb) but with individual choice of threshold for each neuron.

(iv) T is defined based on modified (preprocesses) set of patterns - optimised for the efficacy of the storing/restoring processes.

(iii) Individual thresholds

Adjustable Threshold Network (ATN)

The set of patterns is stored with the Hebb rule.

For each neuron a threshold Φ_i is calculated in the following way:

$$\Phi_i = \frac{1}{2}[S_i^- + S_i^+], \qquad i = 1, \dots, N, \qquad \text{where}$$

$$S_i^- = \max_k \left\{ \sum_{j \neq i} t_{ij} x_j^k \mid 1 \leq k \leq M,\, x_i^k = -1 \right\},$$

$$S_i^+ = \min_k \left\{ \sum_{j \neq i} t_{ij} x_j^k \mid 1 \leq k \leq M,\, x_i^k = +1 \right\}.$$

ATN Method

Threshold Θ_i is defined as the mean value of the extreme input values for the $\boldsymbol{i}$ – th neuron generated in the two following sets: X_i^+, X_i^-

$$X_i^+ \text{ is the set of } X^k, \text{ for which } x_i^k = +1$$

$$X_i^- \text{ is the set of } X^k, \text{ for which } x_i^k = -1$$

Rule non-local, offline

Compared to Hebb rule the capacity increases:

- for uniform vectors by about 25%
- for rare vectors (p=0.3) about 2-3-times!

Thank you!