

شبکه‌های ارقابندی

شبکه‌های عمیق مصنوعی

۰۱-۷۱۳-۱۱-۱۳

بخش پنجم



دانشگاه شهید بهشتی

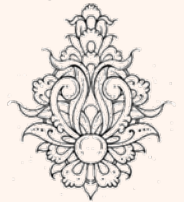
دانشکده‌ی مهندسی برق و کامپیوتر

بهار ۱۳۹۴

احمد محمودی ازناوه

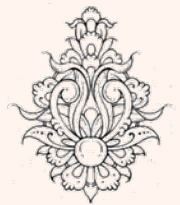
فهرست مطالب

- شبکه‌های رقابتی
 - مقایسه گذرای سه نوع شبکه‌ی مختلف
 - شبکه‌ی همینگ
 - لایه‌ی feedforward
 - لایه‌ی رقابتی – recurrent
 - آموزش رقابتی
 - مشکلات آموزش رقابتی
- شبکه‌های خودسازمان‌ده (SOM)
- LVQ

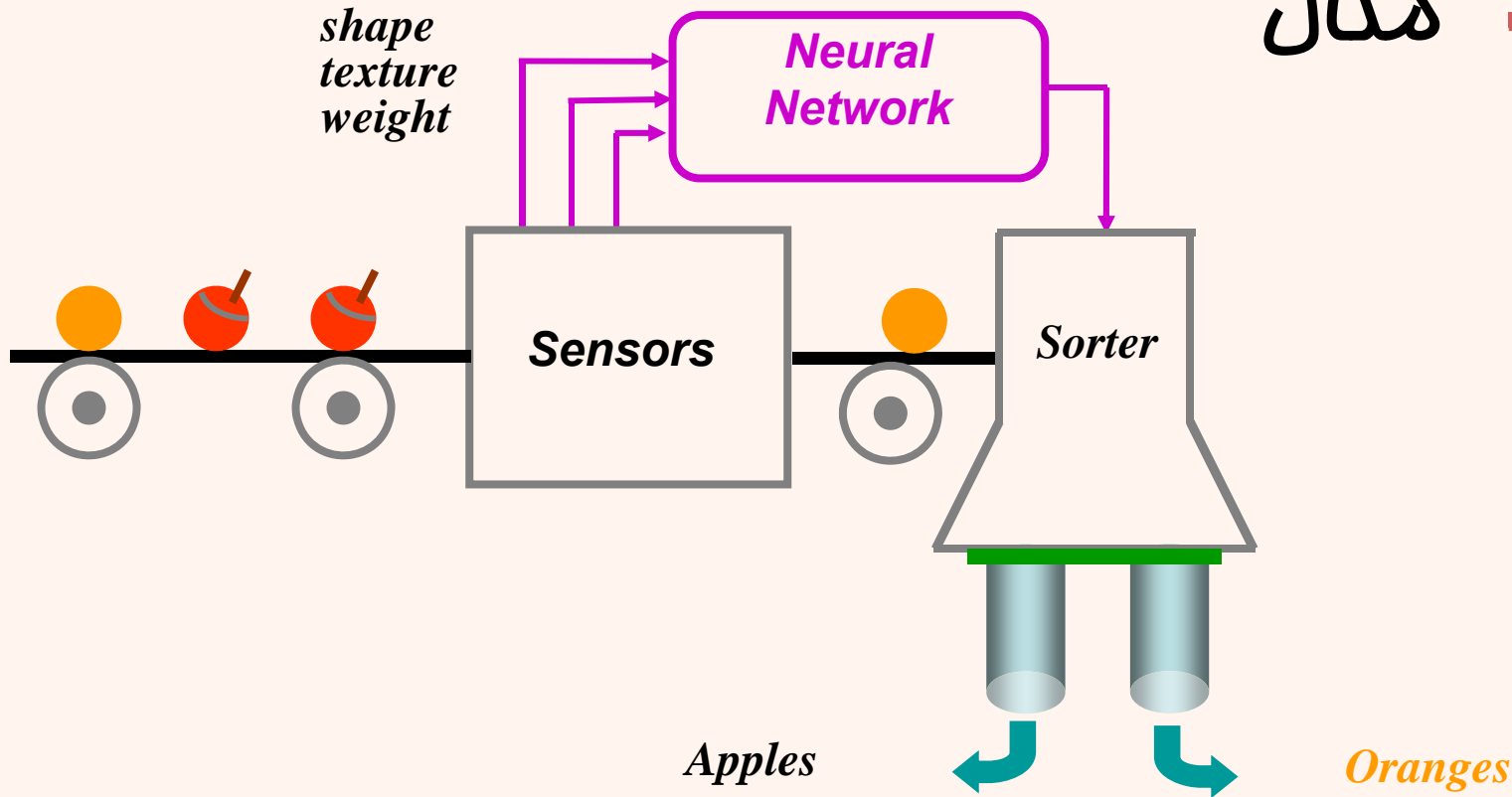


پیش‌گفتار

- برای حل یک مسأله‌ی تشخیص الگو از راه‌حل‌های متفاوتی می‌توان استفاده کرد:
 - Perceptron
 - Hamming network
 - Hopfield network



مثال

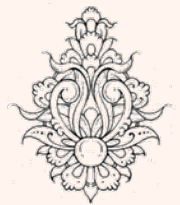


$$\mathbf{p} = \begin{bmatrix} \text{shape} \\ \text{texture} \\ \text{weight} \end{bmatrix} \Rightarrow \mathbf{p}(\text{apple}) = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}, \quad \mathbf{p}(\text{orange}) = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$$

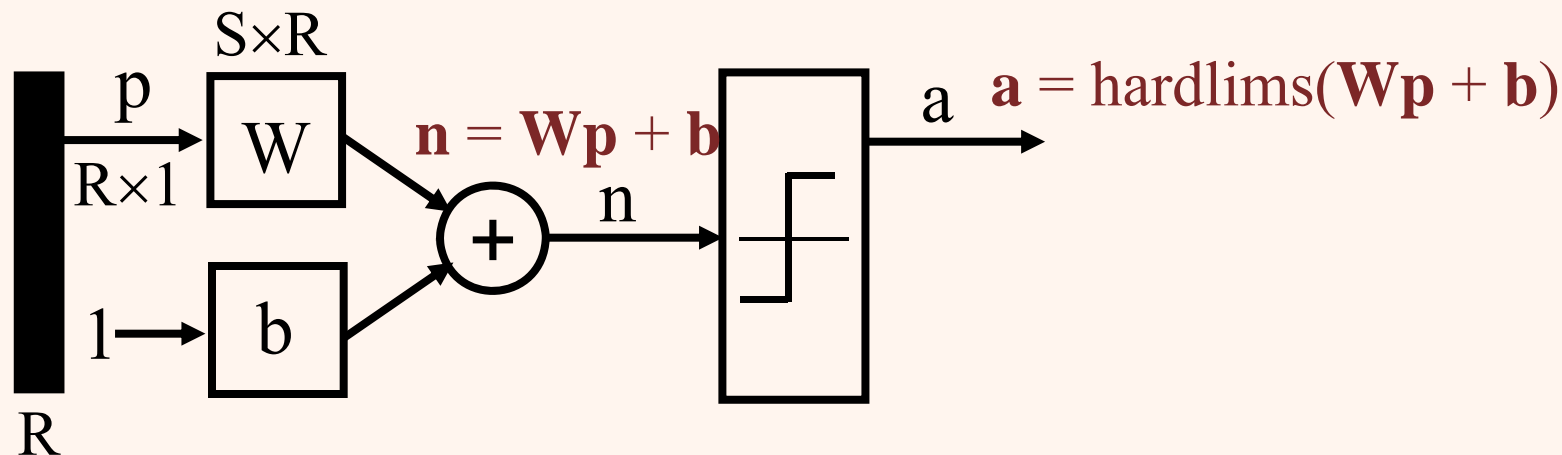
Shape sensor: 1 -- round, -1 -- elliptical.

Texture sensor: 1 -- smooth, -1 -- rough.

Weight sensor: 1 -- > 1 pound, -1 -- < 1 pound.

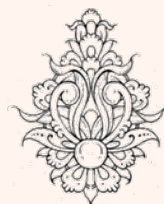


Single-layer Perceptron

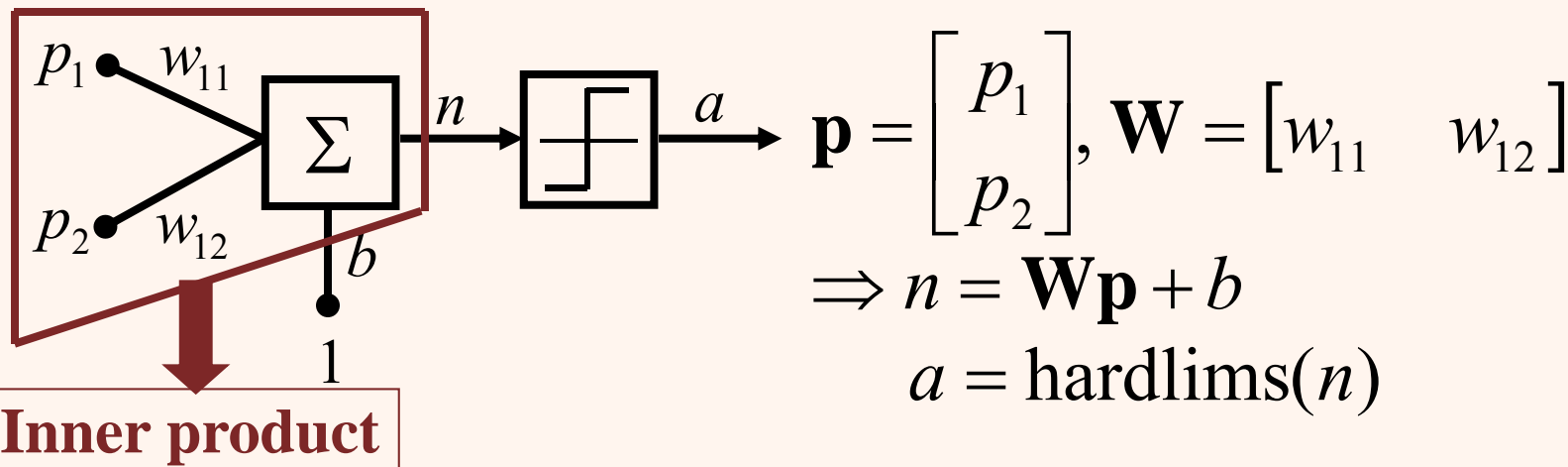


Symmetrical Hard Limit

- $\mathbf{a} = -1$, if $\mathbf{n} < 0$
- $\mathbf{a} = +1$, if $\mathbf{n} \geq 0$
- MATLAB function: `hardlims`



Two-Input / Single-Neuron Perceptron



• یک نرون می‌تواند یک بردار ورودی را به دو گروه دسته‌بندی کند.

If $\mathbf{W}\mathbf{p} \geq -b$,
then

$a = +1$;

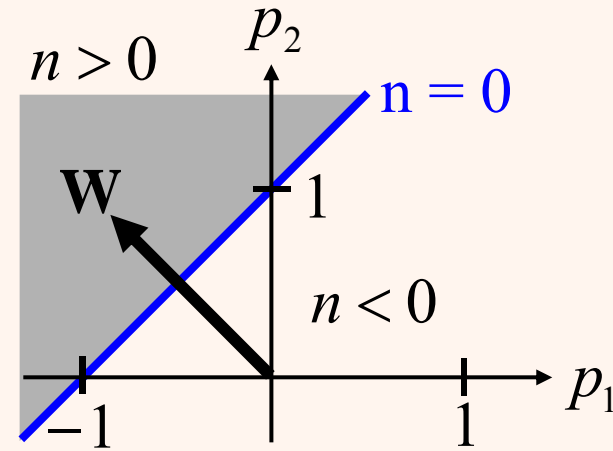
otherwise,

$a = -1$.



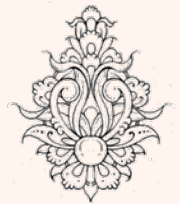
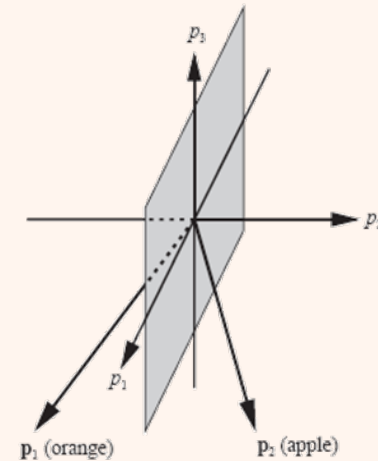
Two-Input / Single-Neuron Perceptron

$$\mathbf{W} = [w_{11} \quad w_{12}] = [-1 \quad 1], b = -1$$
$$\Rightarrow n = \mathbf{W}\mathbf{p} + b = -p_1 + p_2 - 1$$
$$\Rightarrow a = \text{hardlims}(n)$$



$$\mathbf{p} = \begin{bmatrix} \text{shape} \\ \text{texture} \\ \text{weight} \end{bmatrix} \Rightarrow \text{three - dimensional input } (R = 3)$$

$$n = \mathbf{W}\mathbf{p} + b, a = \text{hardlims}(n)$$

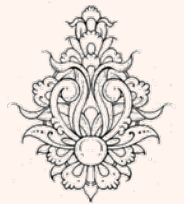


شناسایی الگوها

$$\mathbf{p}(\text{apple}) = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{p}(\text{orange}) = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \Rightarrow \mathbf{W} = [0 \quad 1 \quad 0], \quad b = 0$$

$$\text{Orange : } a = \text{hardlims} \left([0 \quad 1 \quad 0] \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} + 0 \right) = -1$$

$$\text{Apple : } a = \text{hardlims} \left([0 \quad 1 \quad 0] \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} + 0 \right) = 1$$

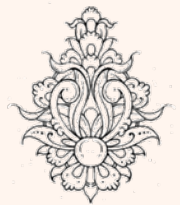


شناسایی الگوها

- حال اگر ورودی به صورتی باشد که تصمیم دقیق میسر نگردد راه حل چیست؟

$$\mathbf{p} = \begin{bmatrix} \textit{shape} \\ \textit{texture} \\ \textit{weight} \end{bmatrix} \Rightarrow \mathbf{p}(\textit{orange}) = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} \Rightarrow \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

$$a = \text{hardlims} \left(\begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + 0 \right) = -1 \Rightarrow \text{orange}$$



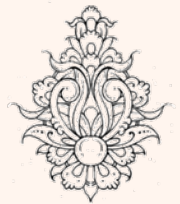
Hamming Network



شبکه‌ی همینگ ساده‌ترین
شبکه‌ی رقابتی است.

Hamming Network

- شبکه‌ی **Hamming** برای دسته‌بندی داده‌های دودویی مطرح شد.
- در شبکه‌ی همینگ دو لایه وجود دارد:
 - لایه‌ی یک (FeedForward)
 - لایه‌ی دو (**Recurrent**)
- لایه‌ی یک
 - تصمیم می‌گیرد ورودی به کدام بردار شناسه نزدیک‌تر است.
- لایه‌ی دو
 - وقتی لایه‌ی دو یا recurrent پایان می‌پذیرد، تنها یک خروجی غیر صفر باقی می‌ماند، در واقع این لایه، لایه‌ی رقابتی است.



Recurrent Network:

شبکه‌ای است که در آن **feedback** وجود دارد؛ در آن برخی خروجی‌ها به لایه‌ی ورودی متصل هستند. خروجی مرحله‌ی قبل به عنوان ورودی مرحله‌ی فعلی استفاده می‌شود.

Hamming Network

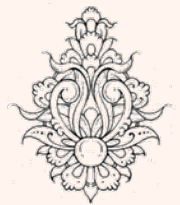
- در طبقه‌بندی همینگ هدف دسته‌بندی ورودی‌ها در S کلاس متفاوت است.
- در شبکه‌ی همینگ کلاس‌های S معلوم هستند (مرکز هر کلاس مشخص است).
- فرض می‌شود S بردار شناسه (Prototype) وجود دارد.

$$P_{1(R \times 1)}, P_{2(R \times 1)}, \dots, P_{S(R \times 1)}$$

بردارهای شناسه

- ورودی: برداری به اندازه‌ی $1 \times R$ است.

$$X = [x_1, \dots, x_R]^T$$



فاصله

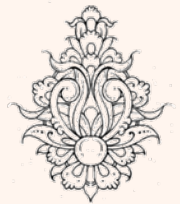
- در خوشه‌بندی و شبکه‌های رقابتی لازم معیاری که بیان‌گر فاصله‌ی ورودی از بردار شناسه باشد، تعریف شود. سه معیار متداول به ترتیب زیر هستند:

$$d(\mathbf{X}, \mathbf{P}) = \sum_k [\mathbf{X}(k) - \mathbf{P}(k)]^2$$

– فاصله‌ی اقلیدسی:

– **ضرب داخلی:** بزرگ‌تر بودن حاصل‌ضرب داخلی به معنای کم‌تر بودن زاویه‌ی دوبردار است.

– **فاصله‌ی همینگ:** برابر با تعداد مؤلفه‌هایی است که با هم تفاوت دارند.



هدف

- طبقه‌بندی نمودن ورودی‌ها در S کلاس مختلف با S بردار شناسه

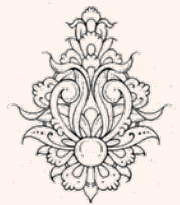
– بردارهای شناسه هم‌اندازه‌ی بردار ورودی است.

- در صورتی که هم‌اندازه نباشد آن‌ها را هم‌اندازه (Normalize) می‌کنیم.

$$X_i = \frac{X_i}{\|X_i\|} \|P_j\|$$

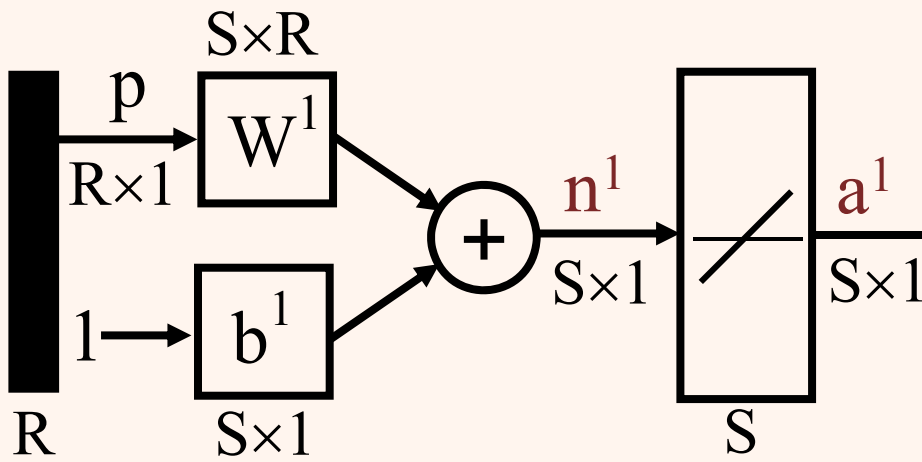
• پس از نرمالایز کردن، می‌توان داده‌ها را به صورت نقاط روی ممیط یک دایره (ایرکله) در نظر گرفت.

$$Y_{S \times 1} = W_{S \times R} X_{R \times 1} + B_{S \times 1}^T$$



Hamming Network

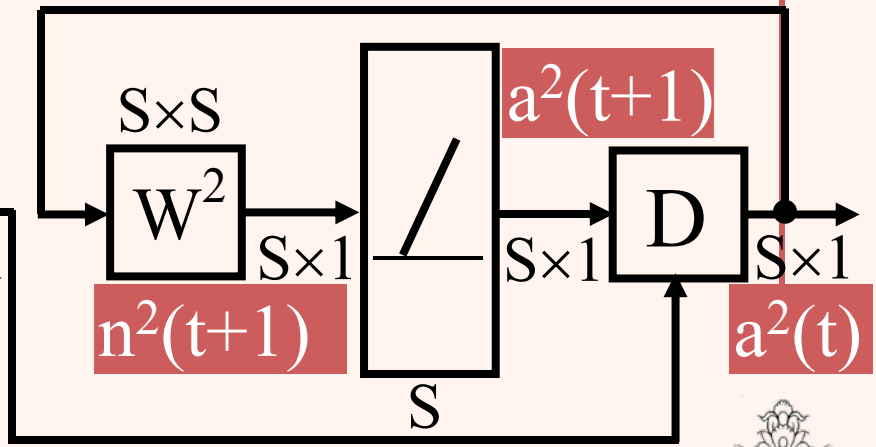
Feedforward Layer



$$n^1 = W^1 p + b^1$$

$$a^1 = \text{purelin}(n^1)$$

Recurrent Layer



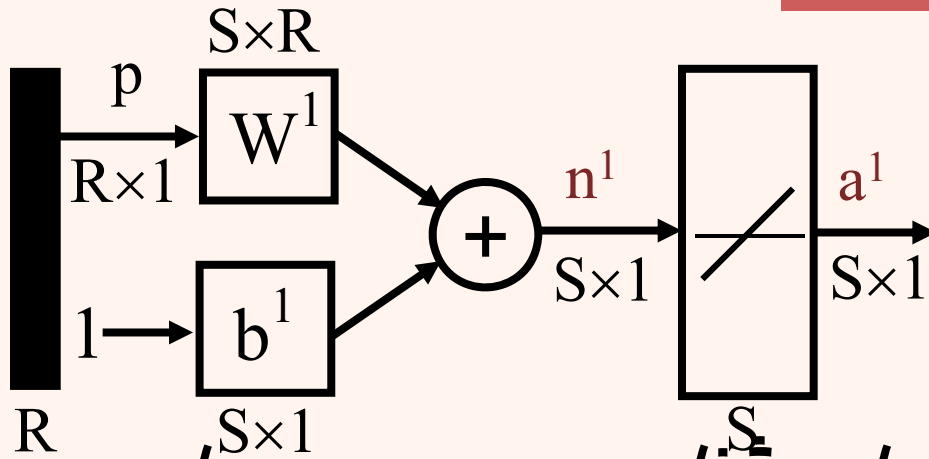
$$n^2(t+1) = W^2 a^2(t)$$

$$a^2(t+1) = \text{poslin}[n^2(t+1)]$$

$$a^2(0) = a^1$$

Feedforward Layer

Hamming Network



• لایه‌ی یک

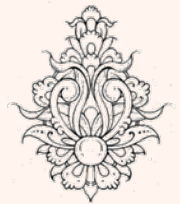
$$n^1 = W^1 p + b^1$$
$$a^1 = \text{purelin}(n^1)$$

• همبستگی بین بردارهای شناسه و ورودی محاسبه می‌شوند.

• ماتریس W را با استفاده از بردارهای شناسه مقداردهی می‌شود.

• بایاس را به اندازه‌ی مثبت (R) در نظر می‌گیریم که خروجی منفی نگردد.

در صورتی که ورودی‌ها همه دو مقداری (1 و -1) باشند، بین ضرب داخلی و فاصله‌ی همینگ چه ارتباطی وجود خواهد داشت؟

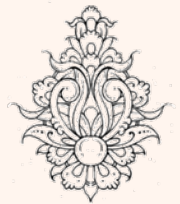


Hamming Network

$$Y_{S \times 1} = W_{S \times R} \cdot X_{R \times 1} + B_{S \times 1}^T$$

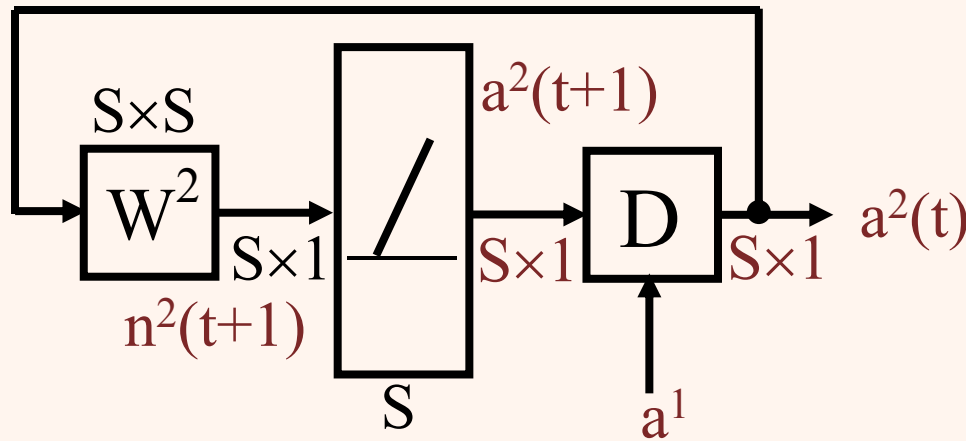
$$Y = \begin{bmatrix} P_1^T \\ \cdot \\ \cdot \\ P_S^T \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ \cdot \\ \cdot \\ x_R \end{bmatrix} + \begin{bmatrix} R \\ \cdot \\ \cdot \\ R \end{bmatrix} \quad \theta_k = \angle[P_k, \mathbf{x}]$$

$$Y = \begin{bmatrix} P_1^T \cdot \mathbf{x} + R \\ \cdot \\ \cdot \\ P_S^T \cdot \mathbf{x} + R \end{bmatrix} = \begin{bmatrix} \|P_1\| \cdot \|\mathbf{x}\| \cos \theta_0 + R \\ \cdot \\ \cdot \\ \|P_s\| \cdot \|\mathbf{x}\| \cos \theta_s + R \end{bmatrix}$$



Recurrent Layer (MaxNet)

Hamming Network

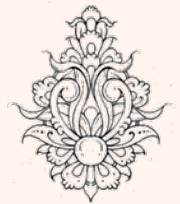


$$\mathbf{n}^2(t+1) = \mathbf{W}^2 \mathbf{a}^2(t)$$

$$\mathbf{a}^2(t+1) = \text{poslin}[\mathbf{n}^2(t+1)]$$

$$\mathbf{a}^2(0) = \mathbf{a}^1$$

- همان لایه‌ی رقابتی “Competitive” است که در این لایه نرون‌ها برای تعیین برنده با هم به رقابت می‌پردازند.
- برنده نرونی است که بیشترین مقدار را دارد. تنها یک نرون برنده خواهد شد.



فرض کنیم فروجی k از همه بزرگتر است چون بقیه از k کمترند اثر تضعیفی بر بقیه بیشتر است و در انتها تنها همین عنصر k غیر صفر باقی خواهد ماند.

$$O_{s \times 1}(0) = y$$

$$O_{s \times 1}(n+1) = f(w_2 \cdot O(n))$$

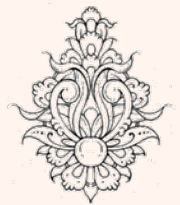
$$W_{(i,j)}^2 = \begin{cases} 1 & i = j \\ -\varepsilon & i \neq j \end{cases} \quad \varepsilon < \frac{1}{s-1}$$

$$f(x) = \begin{cases} x & x \geq 0 \\ 0 & OW \end{cases}$$

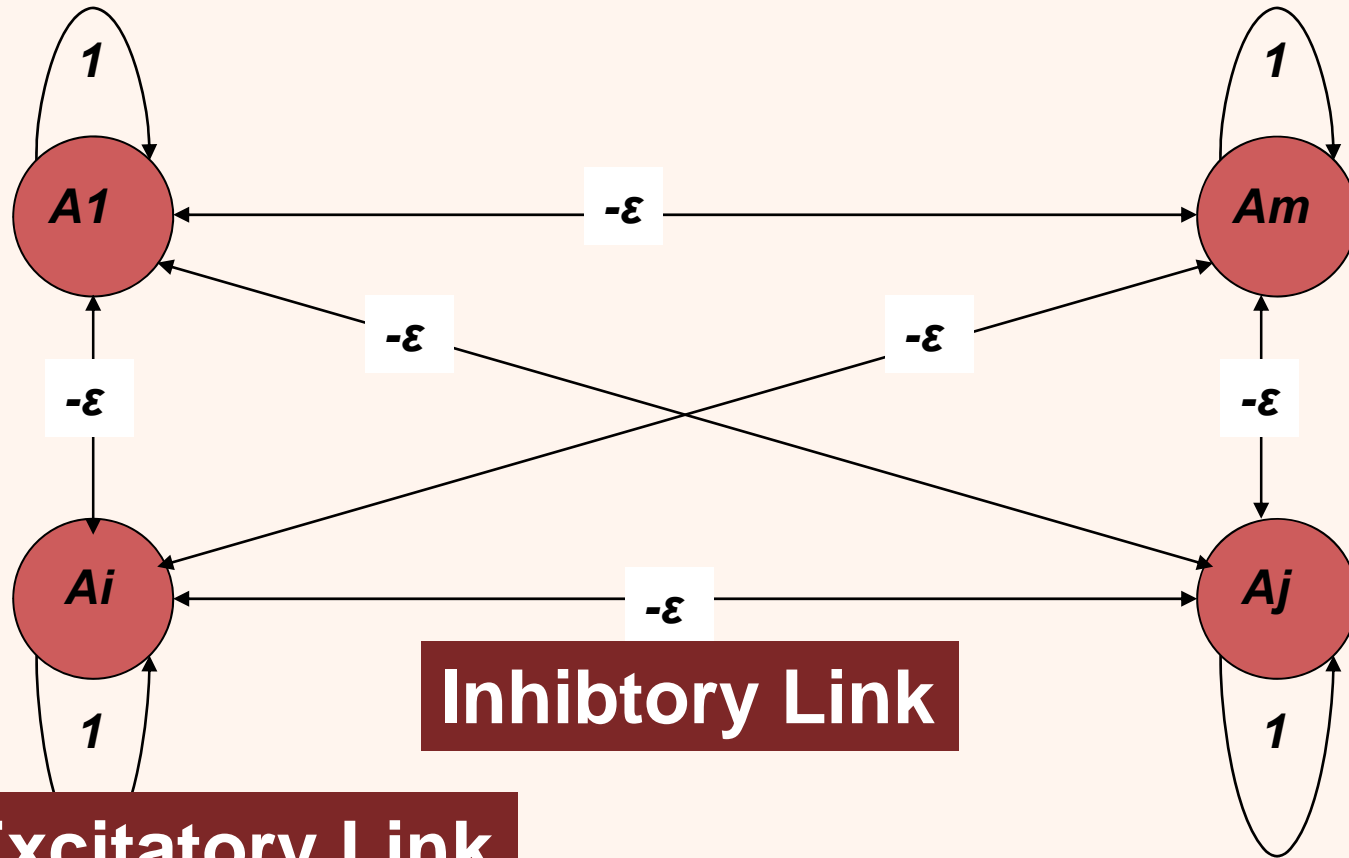
$$W_{2(i,i)} = \begin{bmatrix} 1 & \cdot & \cdot & -\varepsilon \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ -\varepsilon & \cdot & \cdot & 1 \end{bmatrix}$$

Lateral Inhibition: excites itself and inhibits all the other neurons

$$\begin{bmatrix} 1 & \cdot & \cdot & -\varepsilon \\ \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot \\ -\varepsilon & \cdot & \cdot & 1 \end{bmatrix} \begin{bmatrix} O_1(n) \\ \cdot \\ \cdot \\ O_s(n) \end{bmatrix} = \begin{bmatrix} O_1(n) - \varepsilon \sum_{i=2}^s O_i(n) \\ \cdot \\ \cdot \\ O_s(n) - \varepsilon \sum_{i=1}^{s-1} O_i(n) \end{bmatrix}$$



MAXNET



Inhibitory Link

Excitatory Link

Lippmann, R. P. (1987). "An introduction to computing with neural nets." ASSP Magazine, IEEE 4(2): 4-22.



Recurrent Layer

Hamming Network

تعداد نرون ها

$$\mathbf{W}^2 = \begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix}, \mathbf{a}^2(t) = \begin{bmatrix} a_1^2(t) \\ a_2^2(t) \end{bmatrix}, \text{ and } \varepsilon < \frac{1}{S-1}$$

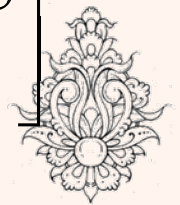
$$\mathbf{a}^2(t+1) = \text{poslin}(\mathbf{W}^2 \mathbf{a}^2(t)) = \text{poslin} \left(\begin{bmatrix} a_1^2(t) - \varepsilon a_2^2(t) \\ a_2^2(t) - \varepsilon a_1^2(t) \end{bmatrix} \right)$$

• تفاوت میان دو عنصری که بزرگ و کوچک هستند در این صورت افزایش می یابد.

$$\text{If } \mathbf{p} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \Rightarrow \mathbf{a}^2(0) = \mathbf{a}^1 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, \text{ let } \varepsilon = 0.5 \Rightarrow \mathbf{W}^2 = \begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}$$

$$\mathbf{a}^2(1) = \text{poslin} \left(\begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \end{bmatrix} \right) = \text{poslin} \left(\begin{bmatrix} 3 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

$$\mathbf{a}^2(2) = \text{poslin} \left(\begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 0 \end{bmatrix} \right) = \text{poslin} \left(\begin{bmatrix} 3 \\ -1.5 \end{bmatrix} \right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$



مثال

$$\mathbf{W}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 \\ 1 & 1 & -1 \end{bmatrix}, \mathbf{b}^1 = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{a}^1 = \mathbf{W}^1 \mathbf{p} + \mathbf{b}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix} \mathbf{p} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^T \mathbf{p} + 3 \\ \mathbf{p}_2^T \mathbf{p} + 3 \end{bmatrix}$$

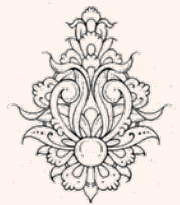
ضرب داخلی دو بردار به علاوه یک بایاس

$$\mathbf{p} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \Rightarrow \mathbf{a}^1 = \mathbf{W}^1 \mathbf{p} + \mathbf{b}^1 = \begin{bmatrix} 1 & -1 & -1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

فاصله همنگ میان دو بردار پودویی برابر است با
تعداد عناصری که در مقایسه با یکدیگر متفاوتند

Output = 2 × (R – Hamming distance)

$$\mathbf{a}^1_1 = 2 \times (3 - 1) = 4, \mathbf{a}^1_2 = 2 \times (3 - 2) = 2$$



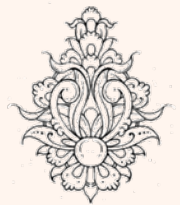
مثال

$$\mathbf{a}^2(1) = \text{poslin} \left(\begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \end{bmatrix} \right) = \text{poslin} \left(\begin{bmatrix} 3 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

$$\mathbf{a}^2(2) = \text{poslin} \left(\begin{bmatrix} 1 & -\frac{1}{2} \\ -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 0 \end{bmatrix} \right) = \text{poslin} \left(\begin{bmatrix} 3 \\ -1.5 \end{bmatrix} \right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix}$$

- هنگامی که نتیجه‌ی دو تکرار یکسان شود گوییم الگوریتم به همگرایی رسیده است.

در این حالت بردارشناسی اول یعنی پرتقال به عنوان برنده در نظر گرفته شده است



مثال

- در یک فضای سه بعدی بردارهای شناسه مانند زیر است ورودی اگر x باشد:

$$p_1 = [1 \quad -1 \quad -1]^T \quad p_2 = [1 \quad 1 \quad -1]^T$$

$$x = [2 \quad 3 \quad -6]^T$$

$$b^1 = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

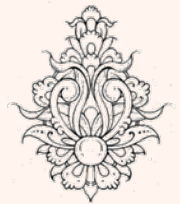
- ابتدا بایاس را در نظر می‌گیریم:

- بردار ورودی را نرمالیزه می‌کنیم:

$$\|p_1\|^2 = \|p_2\|^2 = 3 \quad \|x\|^2 = 49$$

$$\|x_{new}\|^2 = \frac{49}{a} = 3 \quad \|x_{new}\|^2 = \frac{4}{a} + \frac{9}{a} + \frac{36}{a}$$

$a \approx 16$



ادامہی مثال

$$\|x_{new}\|^2 = \frac{4}{a} + \frac{9}{a} + \frac{36}{a}$$

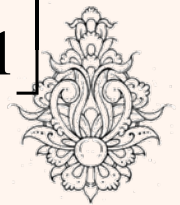
$$x_{new} = \left[\frac{2}{\sqrt{a}}, \frac{3}{\sqrt{a}}, \frac{-6}{\sqrt{a}} \right]$$

$$x_{new} = [0.5, 0.71, -1.5]$$

$$x = \begin{bmatrix} 0.5 \\ 0.71 \\ -1.5 \end{bmatrix} \Rightarrow \mathbf{a}^1 = \mathbf{W}^1 x + \mathbf{b}^1 = \begin{bmatrix} 1 & -1 & -1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.71 \\ -1.5 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1.29 \\ 2.71 \end{bmatrix}$$

$$\mathbf{W}^2 = \begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix}, \quad \varepsilon < \frac{1}{S-1}$$
$$O_{s \times 1}(0) = y$$

$$O_{s \times 1}(n+1) = f(w_2 \cdot O(n))$$



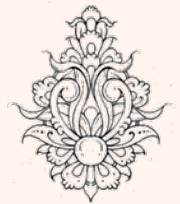
$$O_{s \times 1}(0) = y$$

$$O_{s \times 1}(n + 1) = f(w_2 \cdot O(n))$$

$$O_{s \times 1}(1) = f(w_2 \cdot O(0)) = f\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix} \begin{bmatrix} 1.29 \\ 2.71 \end{bmatrix}\right) = f\left(\begin{bmatrix} < 0 \\ 2.07 \end{bmatrix}\right)$$

- اگر ϵ کوچک در نظر گرفته شود سرعت همگرایی پایین می‌آید.

$$W^2 = \begin{bmatrix} 1 & -\epsilon \\ -\epsilon & 1 \end{bmatrix}, \quad \epsilon < \frac{1}{S-1}$$



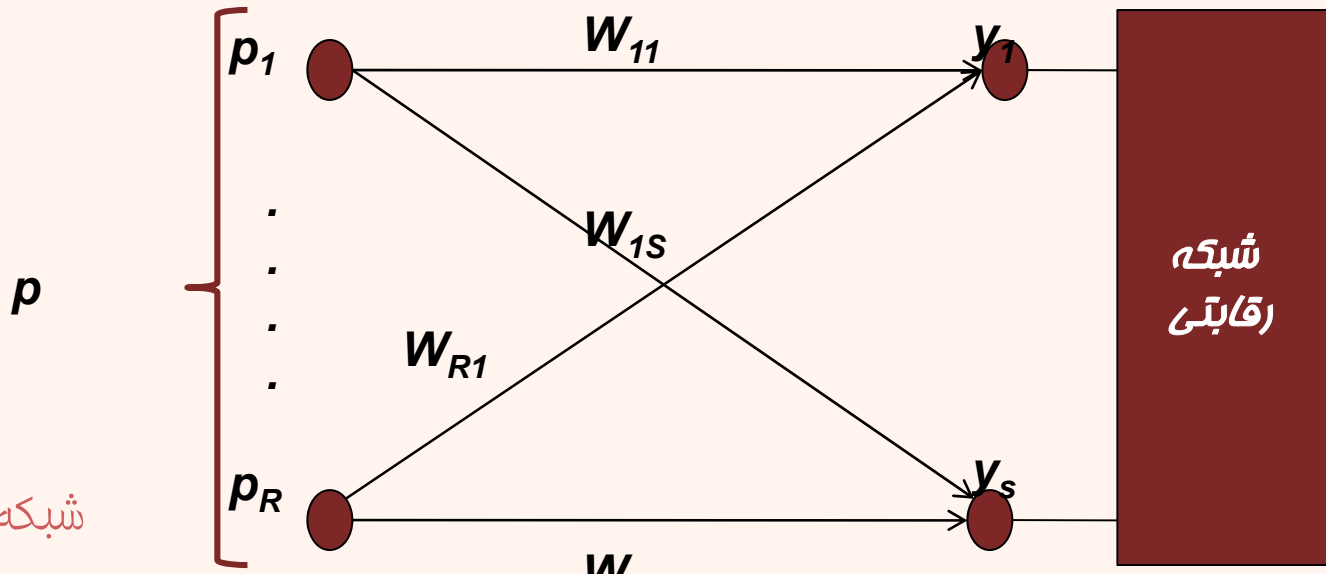
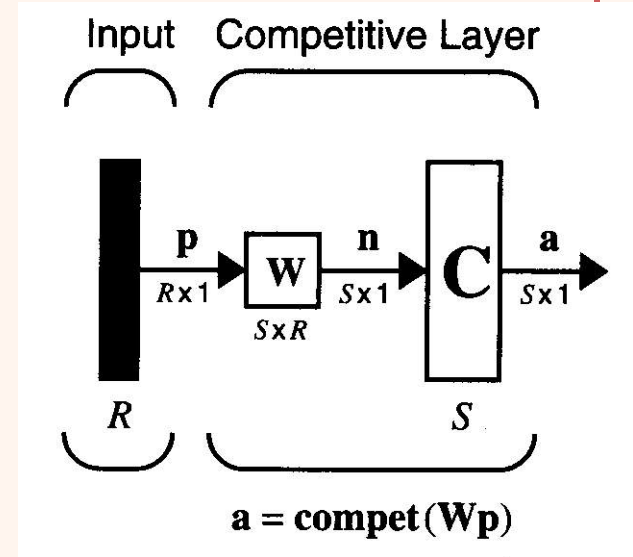
A recurrent competitive layer

$$\mathbf{n} = \mathbf{W}\mathbf{p} = \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_S^T \end{bmatrix} \mathbf{p} = \begin{bmatrix} L^2 \cos \theta_1 \\ L^2 \cos \theta_2 \\ \vdots \\ L^2 \cos \theta_S \end{bmatrix}$$

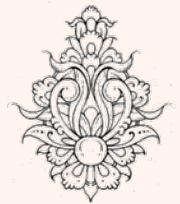
$$n_{i^*} = \max \{n_1, n_2, \dots, n_S\}$$

$$\mathbf{a} = \text{compet}(\mathbf{n}) \quad a_i = \begin{cases} 1, & i = i^* \\ 0, & i \neq i^* \end{cases}$$

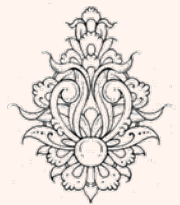
لایه رقابتی



شبکه عصبی



- تنها تعداد دسته‌ها مشخص است.
- بردارهای شناسه ناشناخته‌اند.
- ماتریس وزن‌ها تصادفی انتخاب می‌شود.
- هدف:
- یافتن ماتریس وزن (بردارهای شناسه) بهینه
- می‌توان هر سطر ماتریس W به صورت تصادفی انتخاب کرد.
- ورودی‌ها به اندازه‌ی $R \times 1$
- تعداد گروه‌ها به اندازه‌ی S



آموزش رقابتی (ادامه...)

مراحل الگوریتم:

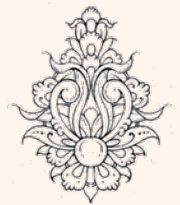
- W_i ها به طور تصادفی انتخاب می‌شوند.
- همگی به یک مقدار نرمالیزه می‌شوند.
- در هر تکرار یک ورودی به شبکه اعمال می‌شود.

$$P_{(n)} = [P_1, \dots, P_R]$$

- در خروجی رابطه‌ی زیر محاسبه می‌شود:

$$y_{i(n)} = P_{(n)}^T w_{i(n)} = \|P_{(n)}\| \|w_{i(n)}\| \cdot \cos \theta_{i(n)}$$

$$i = 1, 2, \dots, S$$

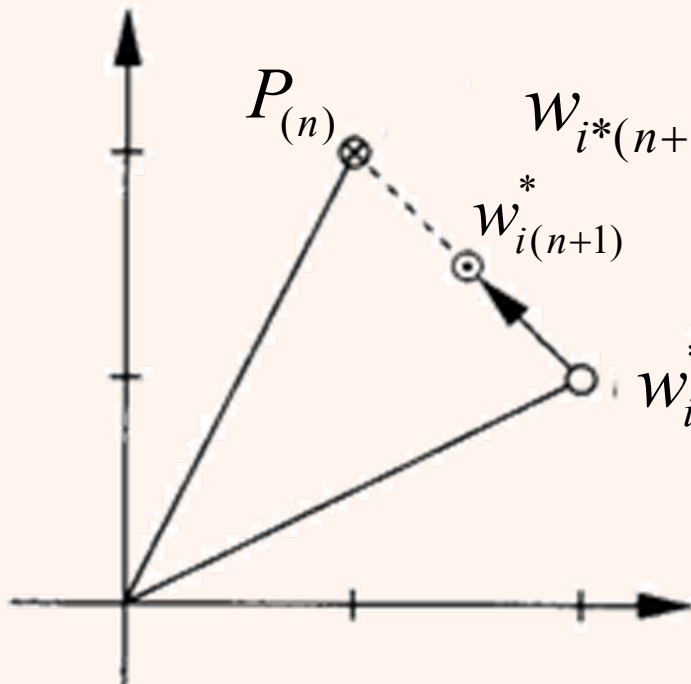


آموزش رقابتی (ادامه...)

- واحدی که دارای بیشترین مقدار فروجی باشد به عنوان **برنده** انتخاب می‌شود و با نام واحد شماره‌ی i^* خوانده می‌شود.

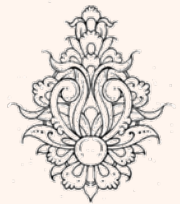
$$y_{i^*} > y_i$$

- مرحله‌ی تطبیق وزن‌ها:



$$w_{i^*(n+1)} = w_{i^*(n)} + \eta [P_{(n)} - w_{i^*(n)}]$$

- اعمال ورودی بعدی و تکرار
- مرحله ۳ به بعد

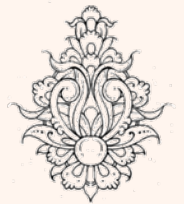
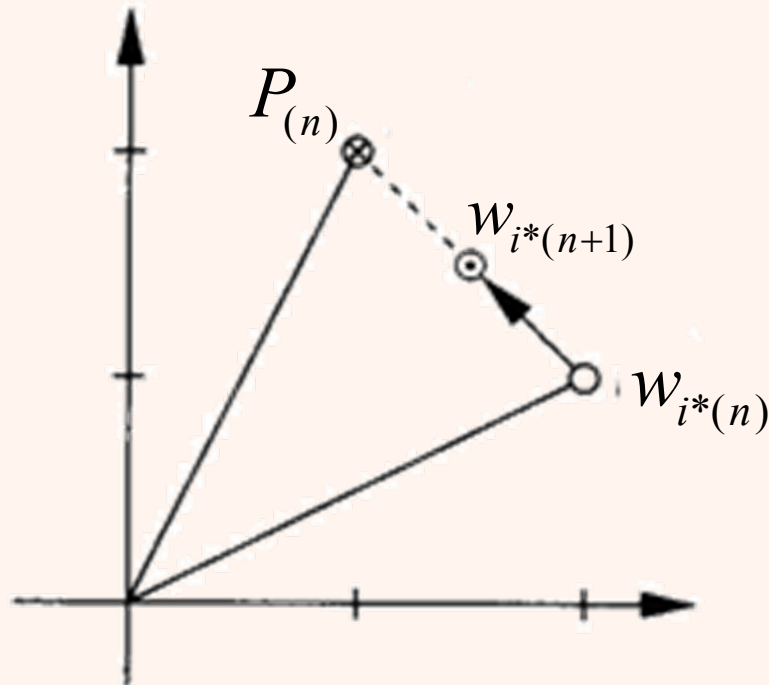


competitive learning rule ۳۰

آموزش رقابتی (ادامه...)

$$w_{i^*(n+1)} = w_{i^*(n)} + \eta [P_{(n)} - w_{i^*(n)}]$$

$$w_{i^*(n+1)} = (1 - \eta) w_{i^*(n)} + \eta P_{(n)}$$



مثال

$$P_1 = \begin{bmatrix} -0.19 \\ 0.98 \end{bmatrix}$$

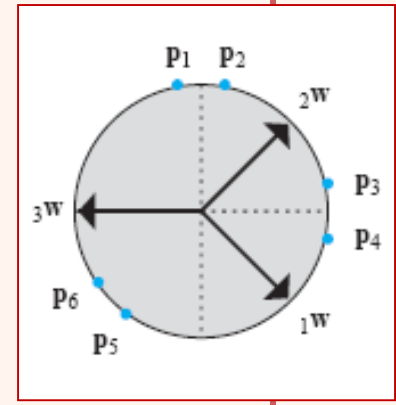
$$P_3 = \begin{bmatrix} 0.98 \\ 0.19 \end{bmatrix}$$

$$P_5 = \begin{bmatrix} -0.58 \\ 0.81 \end{bmatrix}$$

$$P_2 = \begin{bmatrix} 0.19 \\ 0.98 \end{bmatrix}$$

$$P_4 = \begin{bmatrix} 0.98 \\ -0.19 \end{bmatrix}$$

$$P_6 = \begin{bmatrix} -0.81 \\ -0.58 \end{bmatrix}$$



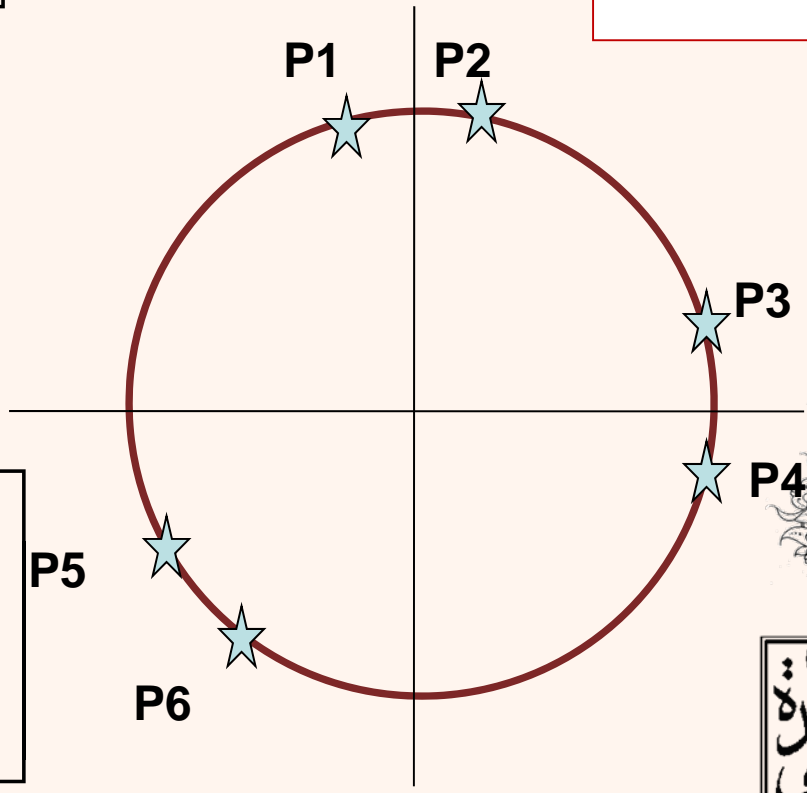
$$W_1 = \begin{bmatrix} 0.7 \\ -0.7 \end{bmatrix}$$

$$W = \begin{bmatrix} W_1^T \\ W_2^T \\ W_3^T \end{bmatrix}$$

$$W_2 = \begin{bmatrix} 0.7 \\ 0.7 \end{bmatrix}$$

$$W = \begin{bmatrix} 0.7 & -0.7 \\ 0.7 & 0.7 \\ -1 & 0 \end{bmatrix}$$

$$W_3 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$



مشخص است که سه گروه داریم وزن‌ها را تصادفی انتخاب می‌کنیم

ادامه‌ی مثال

- ورودی P_2 را به شبکه اعمال می‌کنیم.

$$W.P_2 = \begin{bmatrix} 0.7 & -0.7 \\ 0.7 & 0.7 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 0.19 \\ 0.98 \end{bmatrix} = \begin{bmatrix} -0.55 \\ 0.8 \\ -0.1 \end{bmatrix}$$

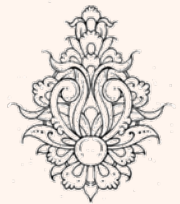
واحد برنده شماره‌ی ۲ است

- وزن واحد برنده را به روز می‌نماییم:

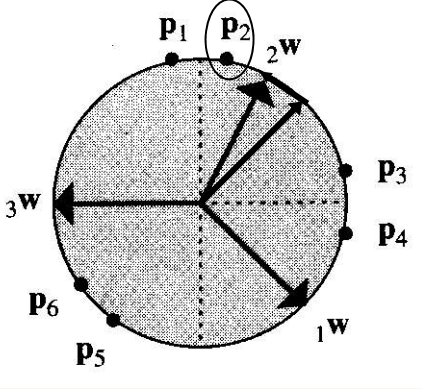
$$w_{2(n+1)} = w_{2(n)} + \eta [P_2 - w_{2(n)}]$$

$$= \begin{bmatrix} 0.7 \\ 0.7 \end{bmatrix} + 0.5 \left[\begin{bmatrix} 0.19 \\ 0.98 \end{bmatrix} - \begin{bmatrix} 0.7 \\ 0.7 \end{bmatrix} \right] = \begin{bmatrix} 0.45 \\ 0.84 \end{bmatrix}$$

$$\begin{bmatrix} 0.7 & -0.7 \\ 0.45 & 0.84 \\ -1 & 0 \end{bmatrix}$$

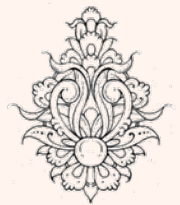
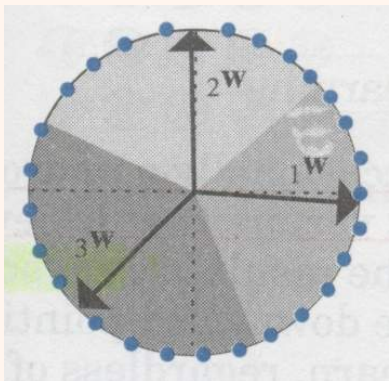
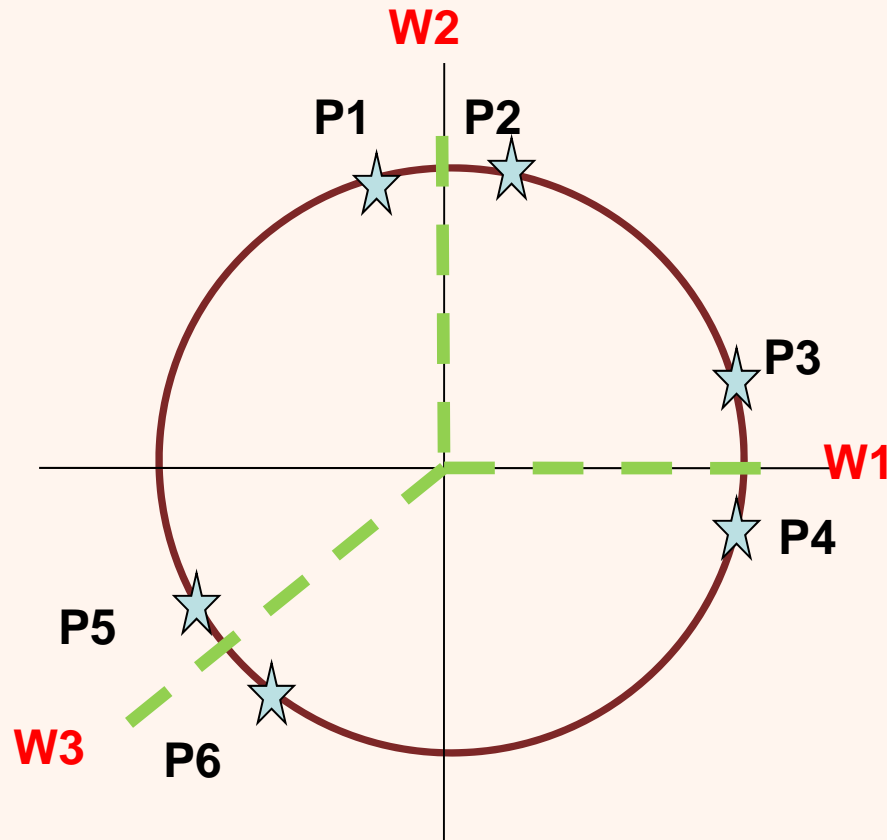


ماتریس وزن به‌روز شده



ادامه‌ی مثال

- ورودی بعدی را اعمال کرده بر حسب واحد برنده وزن‌ها را به روز می‌کنیم.
- در نهایت بردارهای شناسه همانند شکل زیر خواهد شد:



مشکلات یادگیری رقابتی

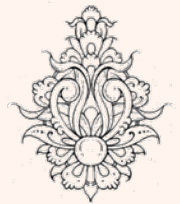
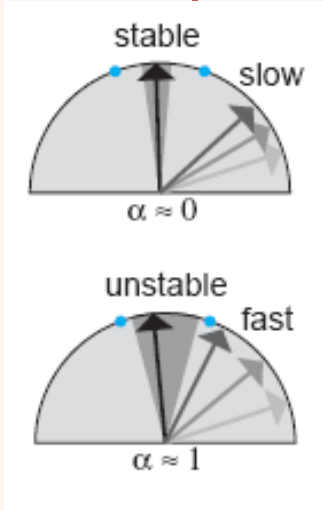
• انتخاب ضریب آموزش η

- $\eta \rightarrow 0$ روند همگرایی را **کند** می‌کند ولی پس از رسیدن به یک بردار شناسه، در آن حالت **پایدار** می‌ماند.

- η بزرگ انتخاب شود، همگرایی **سریع** ولی **ناپایداری** دارد.

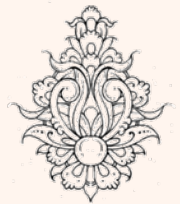
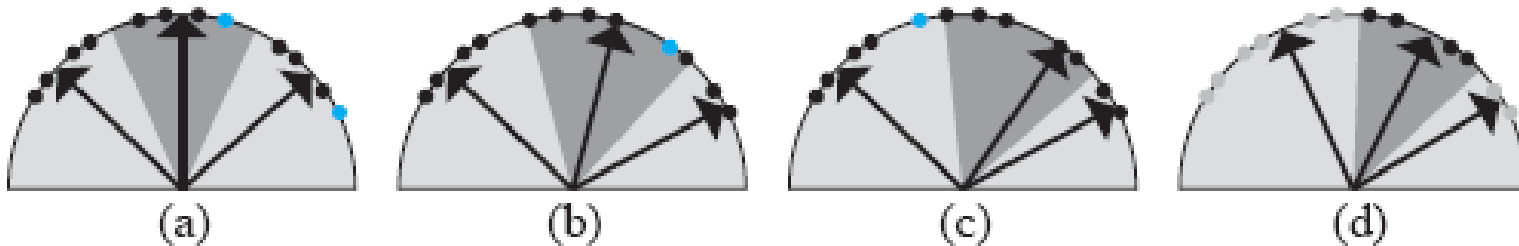
- معمولاً برای رسیدن به مدود اولیه، ابتدا η را بزرگ در نظر می‌گیرند، ولی برای افزایش دقت آن را کاهش می‌دهند.

• در صورتی که نیاز باشد، بردارهای ورودی همواره به روز شوند، این شیوه کارایی نخواهد داشت.



مشکلات یادگیری رقابتی (ادامه...)

- در صورت نزدیک بودن بردارهای شناسه به یکدیگر، جداسازی امکان پذیر نخواهد بود.
- ممکن است بردارهای شناسه به حریج یکدیگر وارد شوند.
- در این حالت فرآیند آموزش **ناپایدار** خواهد بود.

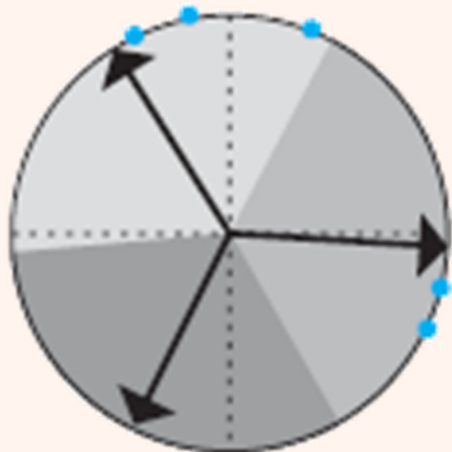


• نرون مرده

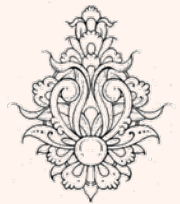
– اگر نرونی خیلی دور باشد هیچگاه برنده نخواهد شد و در نتیجه آموزش نخواهد دید. چنین نرونی را نرون مرده می‌گویند.

– یگ راه برای غلبه بر این مشکل این است که سازوکاری در پیش گرفته شود که شانس نرون‌های برنده برای برنده شدن دوباره کم‌تر شود.

– این کار با دفزودن مقدار منفی به بایاس آن‌ها امکان‌پذیر است. این شیوه «وجدان» خوانده می‌شود.



شبکه عصبی



conscience

مشکلات یادگیری رقابتی (ادامه...)

- در صورتی که تعداد کلاس‌ها مشخص نباشد، این شیوه به کار نخواهد آمد.
- این شیوه قابلیت جداسازی کلاس‌های **غیر ممدب** یا گسسته (شامل نواحی گسسته) را ندارد.



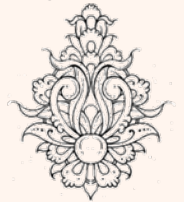
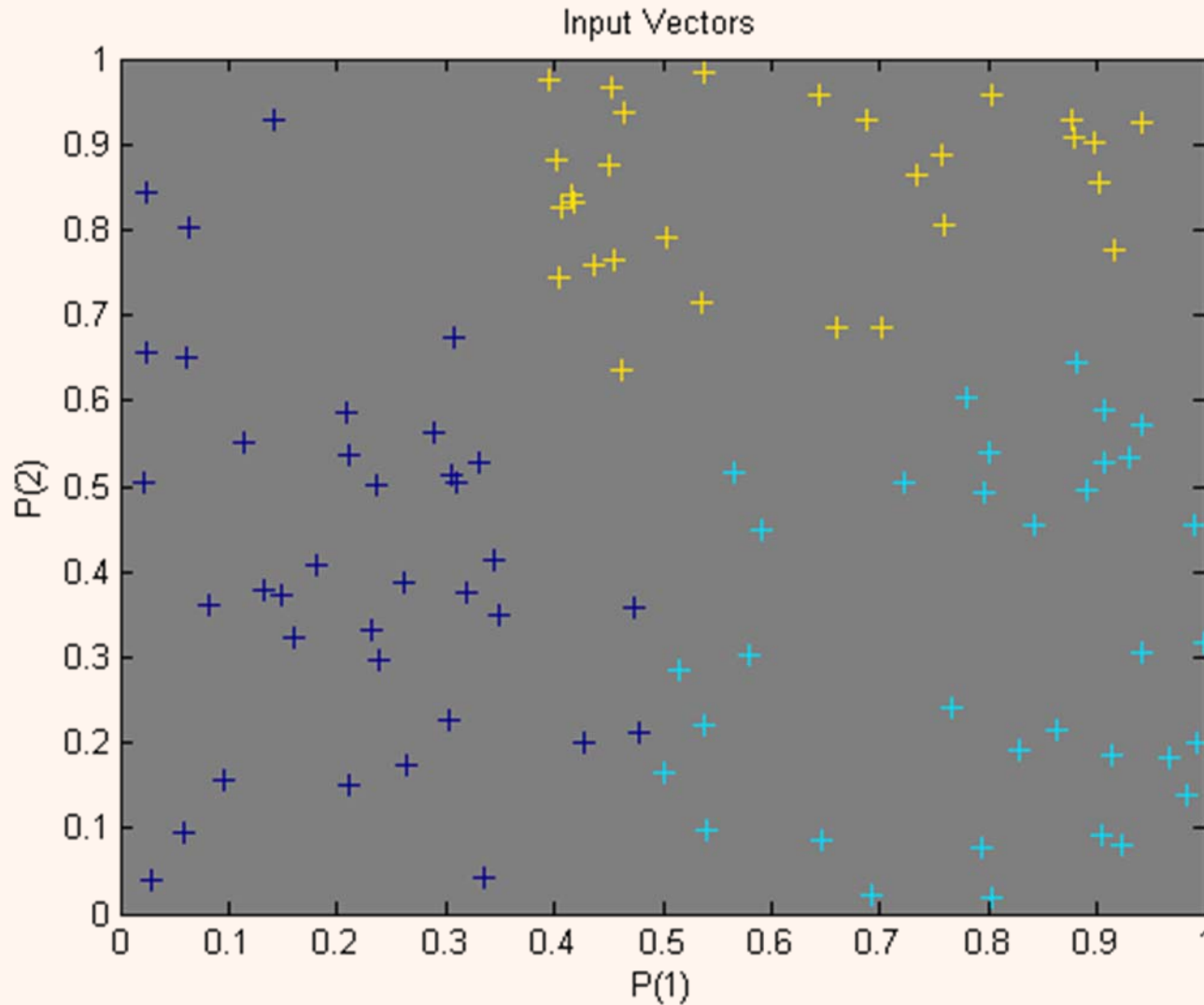
شبکه‌ی رقابتی (مثال)

```
clear all;
P=rand([2 100]);
net = newc([0 1;0 1],3);
net.trainParam.epochs=500;
net = train(net,P);
Y = sim(net,P);
Yc = vec2ind(Y);

plotvec(P,Yc);
title('Input Vectors');
xlabel('P(1)');
ylabel('P(2)');
```



شبکه‌ی رقابتی (ادامه‌ی مثال)



شبکه‌ی رقابتی (مثال)

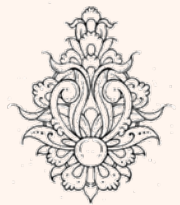
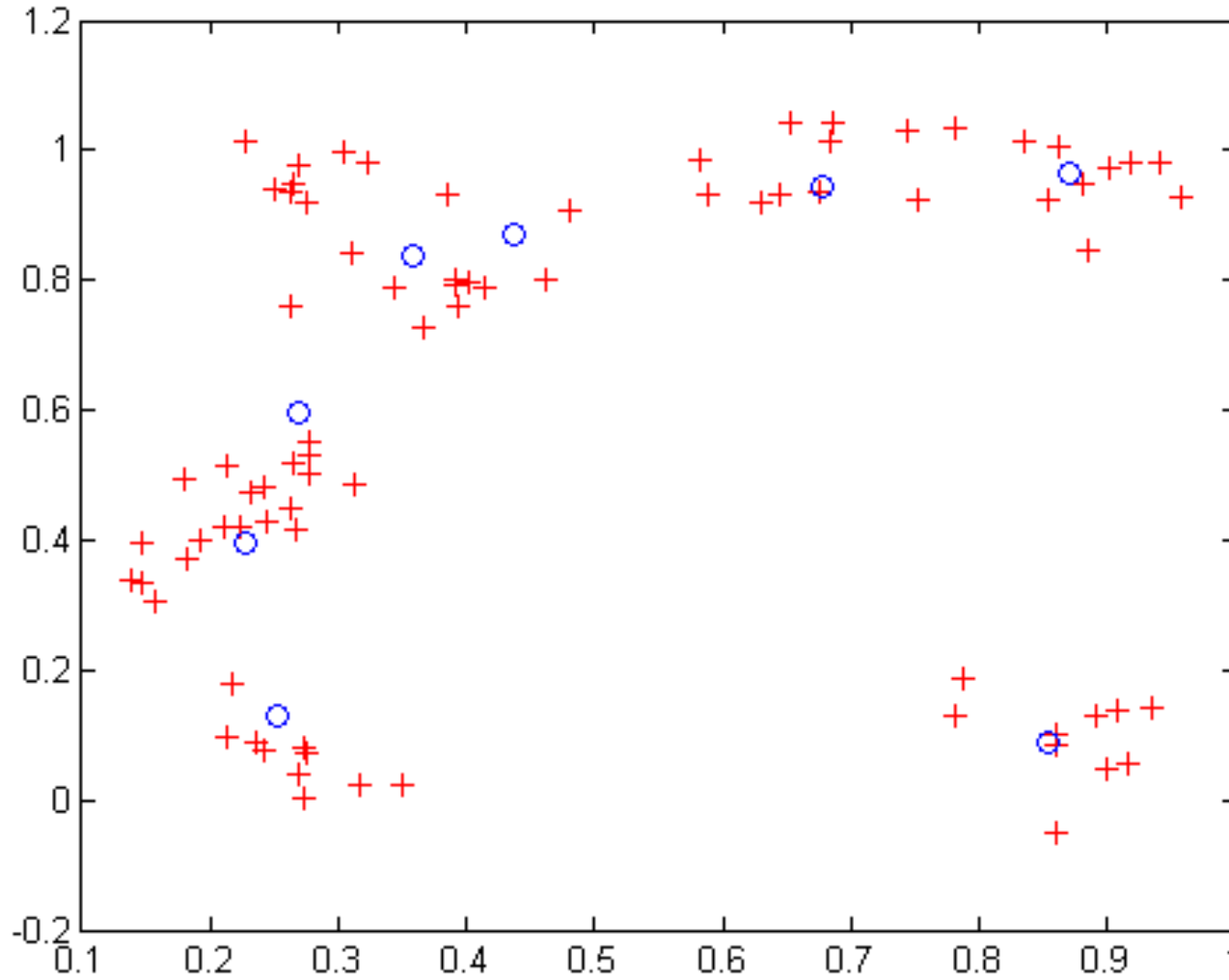
```
% Create P.  
X = [0 1; 0 1]; % Cluster centers to be in these bounds.  
clusters = 8; % This many clusters.  
points = 10; % Number of points in each cluster.  
std_dev = 0.05; % Standard deviation of each cluster.  
P = nngenc(X,clusters,points,std_dev);
```

```
% Plot P.  
plot(P(1,:),P(2:,:),'+r');  
title('Input Vectors');  
xlabel('p(1)');  
ylabel('p(2)');  
net = newc([0 1;0 1],8,.1);  
w = net.IW{1};  
plot(P(1,:),P(2:,:),'+r');  
hold on;  
circles = plot(w(:,1),w(:,2),'ob');
```

```
net.trainParam.epochs =10;  
net = train(net,P);  
w = net.IW{1};  
delete(circles);  
plot(w(:,1),w(:,2),'ob');  
  
p = [0; 0.2];  
a = sim(net,p)
```



شبکه‌ی رقابتی (ادامه‌ی مثال)



SOM

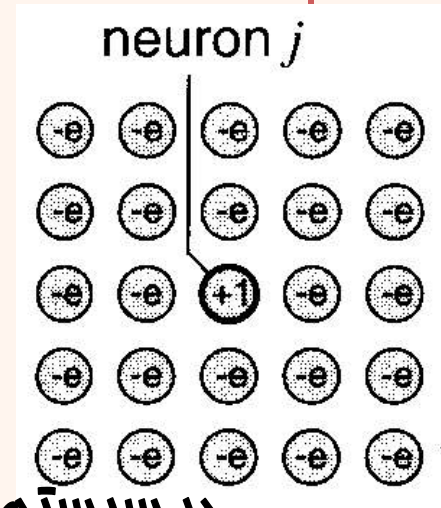


Compet. Layers in Biology

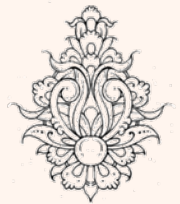
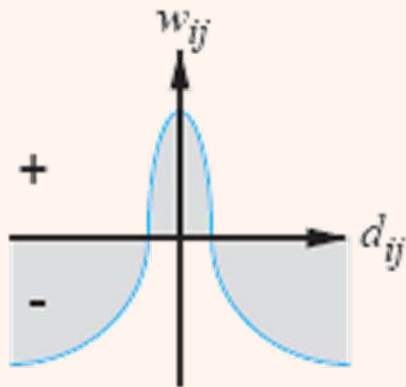
وزن‌های لایه‌ی دوم شبکه‌ی hamming به صورت زیر است:

$$w_{ij} = \begin{cases} 1, & \text{if } i = j \\ -\varepsilon, & \text{if } i \neq j \end{cases} \quad w_{ij} = \begin{cases} 1, & \text{if } d_{ij} = 0 \\ -\varepsilon, & \text{if } d_{ij} \neq 0 \end{cases}$$

Each neuron **reinforces** itself (center), while **inhibiting** all other neurons (surround).



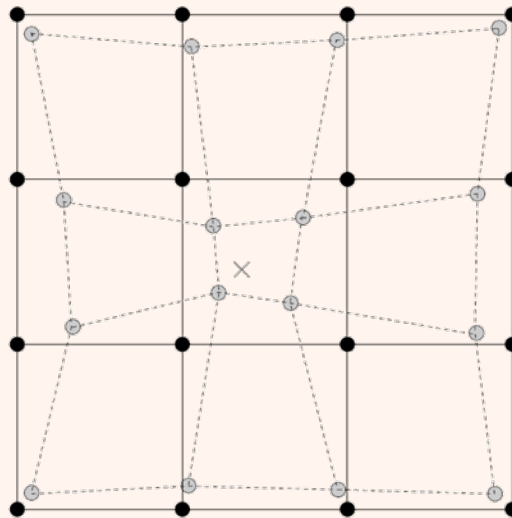
در سیستم‌های بیولوژیکی، بین نامیه‌ی تمرین و تضعیف مرز دقیقی وجود ندارد.



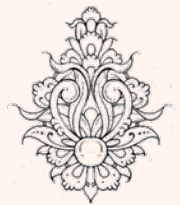
تابع کوره منزلی

SOM (SOFM)

- به نوعی می‌توان گفت تقلید بهتر از شبک‌ی عصبی انسان، منجر به ارائ‌ی SOM توسط kohonen شد.
- در این شبک‌، افزون بر نرون برنده، نرون همسایه‌ی آن هم آموزش می‌بینند.



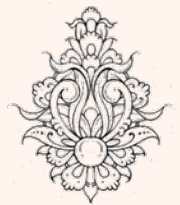
Marc M. Van Hulle



SOM (SOFM)

- تعریف همسایگی و به تبع آن توپولوژی شبکه به طرق مختلف امکان پذیر است.
 - یک بعدی
 - دوبعدی (مشبک دو بعدی)
 - سه بعدی
- این شبکه از انواع شبکه‌های بی‌نظارت (بدون معلم) است.

unsupervised



SOM (SOFM)

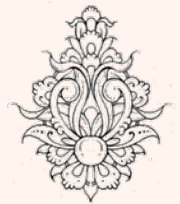
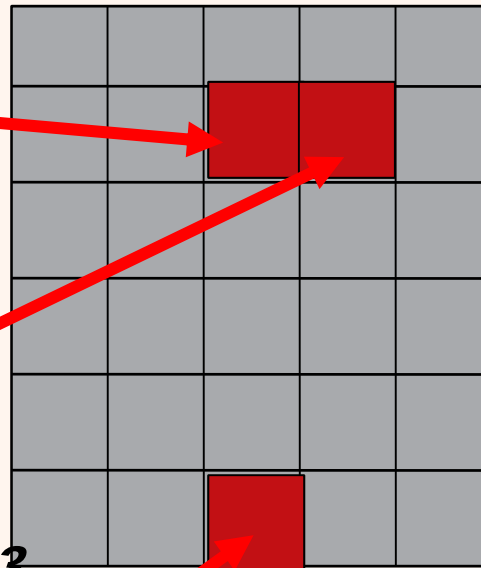
- به نوعی هدف، یافتن مفاهیم مشترک است.



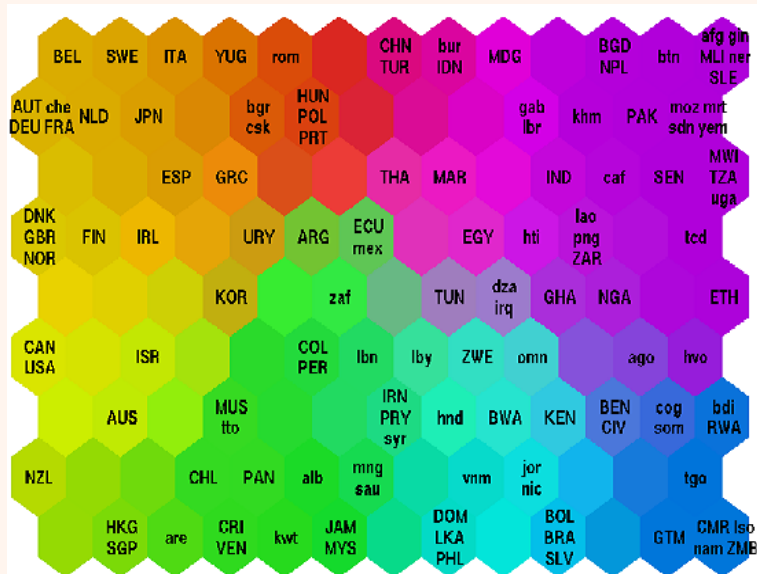
Input Pattern 2



Input Pattern 3

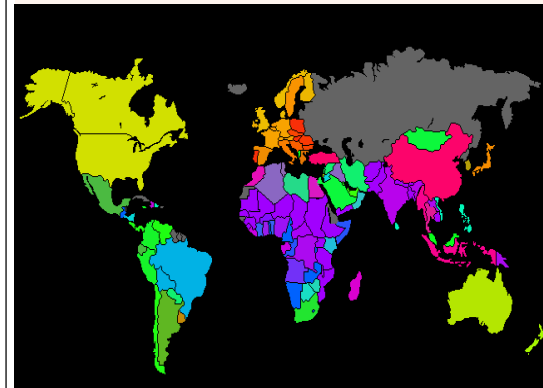


SOM – Result Example



The Country Names

| | | | | | |
|-----|----------------------|-----|--------------------|-----|----------------------------|
| AFG | Afghanistan | OTM | Oman | NZL | New Zealand |
| AGO | Angola | PRG | Prague | OMR | Oman |
| AIB | Algeria | TKO | Tokyo | CHN | China |
| ARE | United Arab Emirates | TTT | Taipei | PAK | Pakistan |
| ARG | Argentina | TKX | Tokyo | PER | Peru |
| ASM | Samoa | IVV | Indonesia | PHL | Philippines |
| ATL | Atlanta | IND | India | PRC | People's Republic of China |
| ATM | Atlanta | IND | India | POL | Poland |
| BDL | Baghdad | IRL | Ireland | PRG | Prague |
| BDR | Bahrain | IRN | Iran, Islamic Rep. | PRY | Paraguay |
| BGR | Bulgaria | IRQ | Iraq | PRT | Portugal |
| BOL | Bolivia | ISR | Israel | ROM | Romania |
| BRA | Brazil | ITA | Italy | RWA | Rwanda |
| BUR | Burkina Faso | JAM | Jamaica | SAL | Saudi Arabia |
| BUR | Burkina Faso | JOR | Jordan | SDK | Sudan |
| BUR | Burkina Faso | JPN | Japan | SGP | Singapore |
| BUR | Burkina Faso | KOR | Korea | SRP | Sri Lanka |
| BUR | Burkina Faso | KTM | Nepal | STL | Sierra Leone |
| BUR | Burkina Faso | KWT | Kuwait | SVK | Slovakia |
| BUR | Burkina Faso | LAO | Laos | SOM | Somalia |
| BUR | Burkina Faso | LBN | Lebanon | SWZ | Switzerland |
| BUR | Burkina Faso | LBR | Liberia | SYR | Syria Arab Rep. |
| BUR | Burkina Faso | LBY | Libya | TCD | Chad |
| BUR | Burkina Faso | LKA | Sri Lanka | TGO | Togo |
| BUR | Burkina Faso | LSD | Lesotho | THA | Thailand |
| BUR | Burkina Faso | MAR | Morocco | TTT | Taipei and Tientsin |
| BUR | Burkina Faso | MDG | Madagascar | TUN | Tunisia |
| BUR | Burkina Faso | MEX | Mexico | TUR | Turkey |
| BUR | Burkina Faso | MFI | Malawi | USA | United States |
| BUR | Burkina Faso | MNI | Mongolia | VEN | Venezuela |
| BUR | Burkina Faso | MOR | Morocco | VNM | Viet Nam |
| BUR | Burkina Faso | MRT | Mauritania | VNM | Viet Nam |
| BUR | Burkina Faso | MUS | Mauritius | VNM | Viet Nam |
| BUR | Burkina Faso | MWI | Malawi | VNM | Viet Nam |
| BUR | Burkina Faso | MYS | Malaysia | VNM | Viet Nam |
| BUR | Burkina Faso | NAM | Namibia | ZAF | South Africa |
| BUR | Burkina Faso | NER | Niger | ZAR | Zaire |
| BUR | Burkina Faso | NGA | Nigeria | ZMB | Zambia |
| BUR | Burkina Faso | NIC | Nicaragua | ZWE | Zimbabwe |
| BUR | Burkina Faso | NLD | Netherlands | | |
| BUR | Burkina Faso | NOR | Norway | | |
| BUR | Burkina Faso | NPL | Nepal | | |

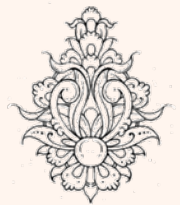
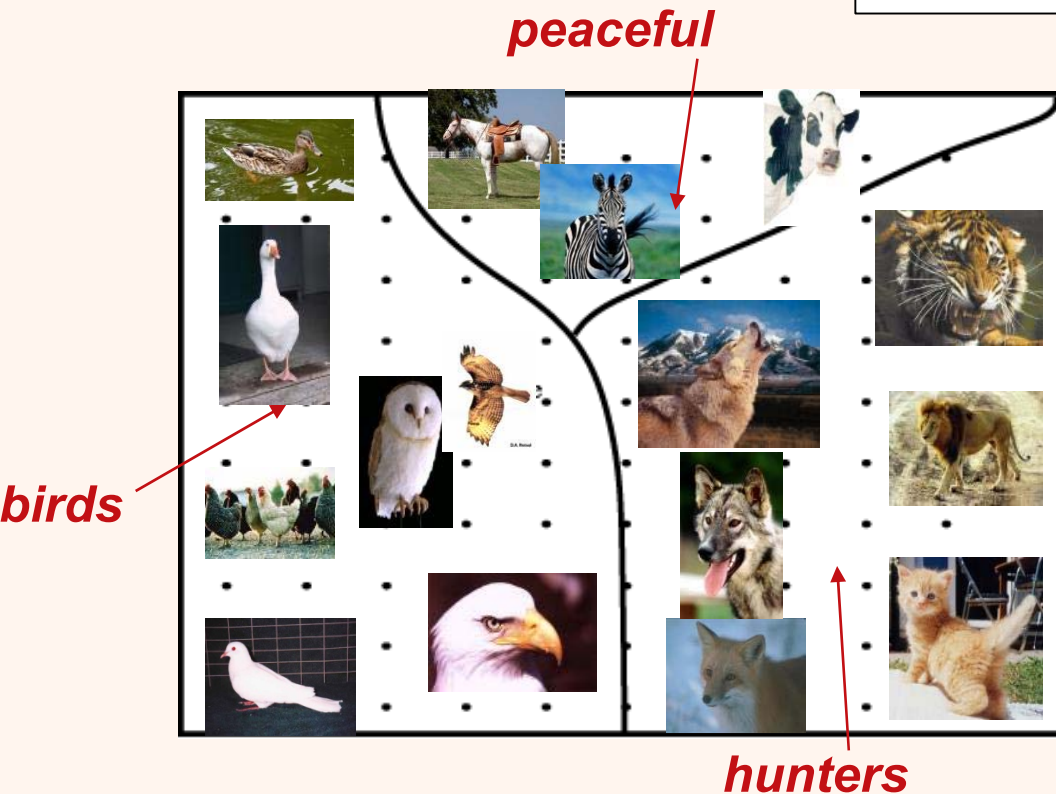


'Poverty map' based on 39 indicators from World Bank statistics (1992)

SOM – Result Example

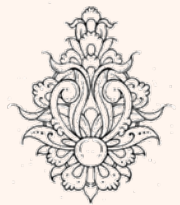
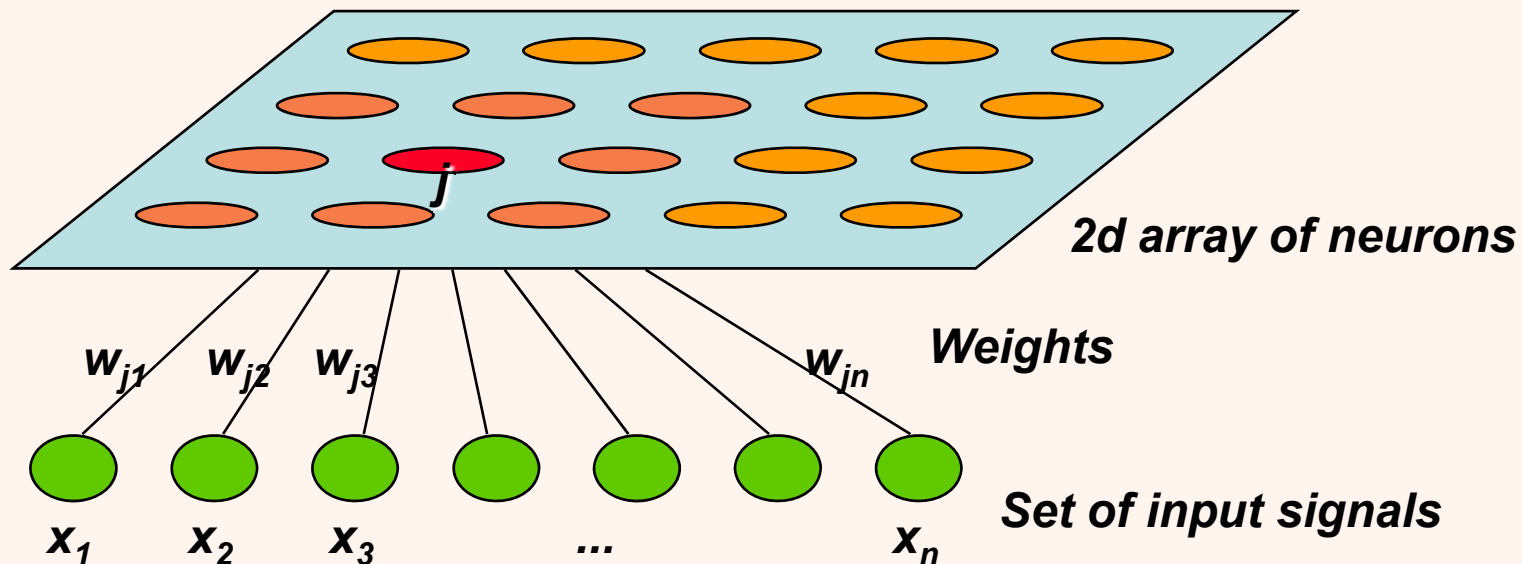
Animal names and their attributes

| | Dove | Hen | Duck | Goose | Owl | Hawk | Eagle | Fox | Dog | Wolf | Cat | Tiger | Lion | Horse | Zebra | Cow |
|-----------------|--------|-----|------|-------|-----|------|-------|-----|-----|------|-----|-------|------|-------|-------|-----|
| is | Small | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | Medium | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Big | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| has | 2legs | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | 4legs | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | Hair | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | Horns | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| | Mare | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| Teeths | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| likes to | Hrt | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| | Run | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| | Fly | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Swim | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

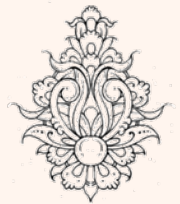
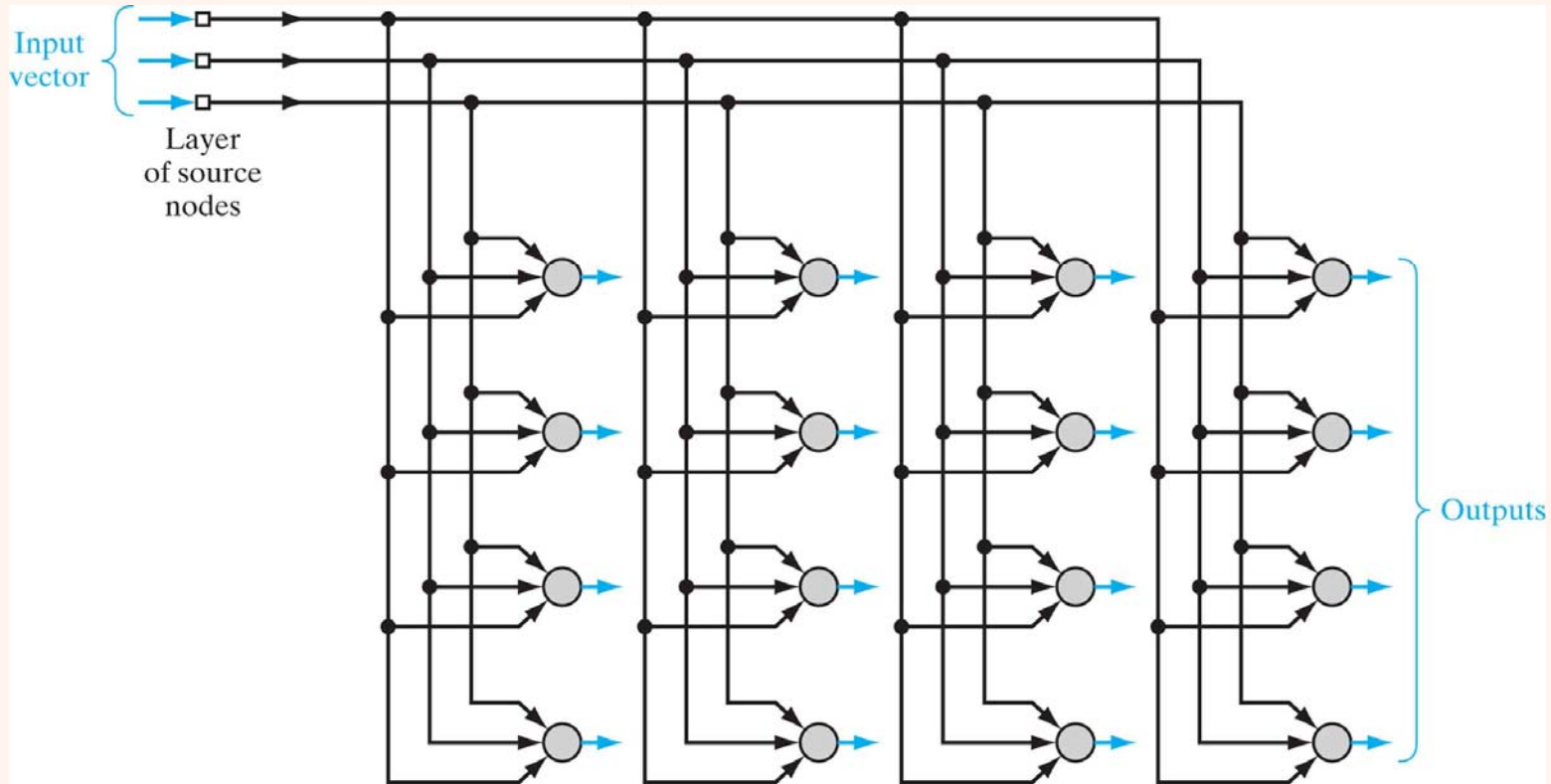


توپولوژی دوبعدی

- در شبکه‌ی SOM با توپولوژی دوبعدی فرض بر این است که مشبکی (Lattice) از نرون‌ها به سیگنال‌های ورودی به شبکه پاسخ می‌دهند.

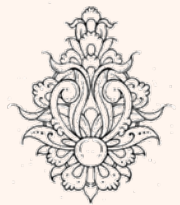


توپولوژی دوبعدی (ادامه...)



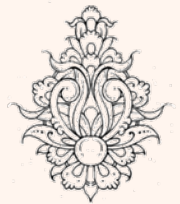
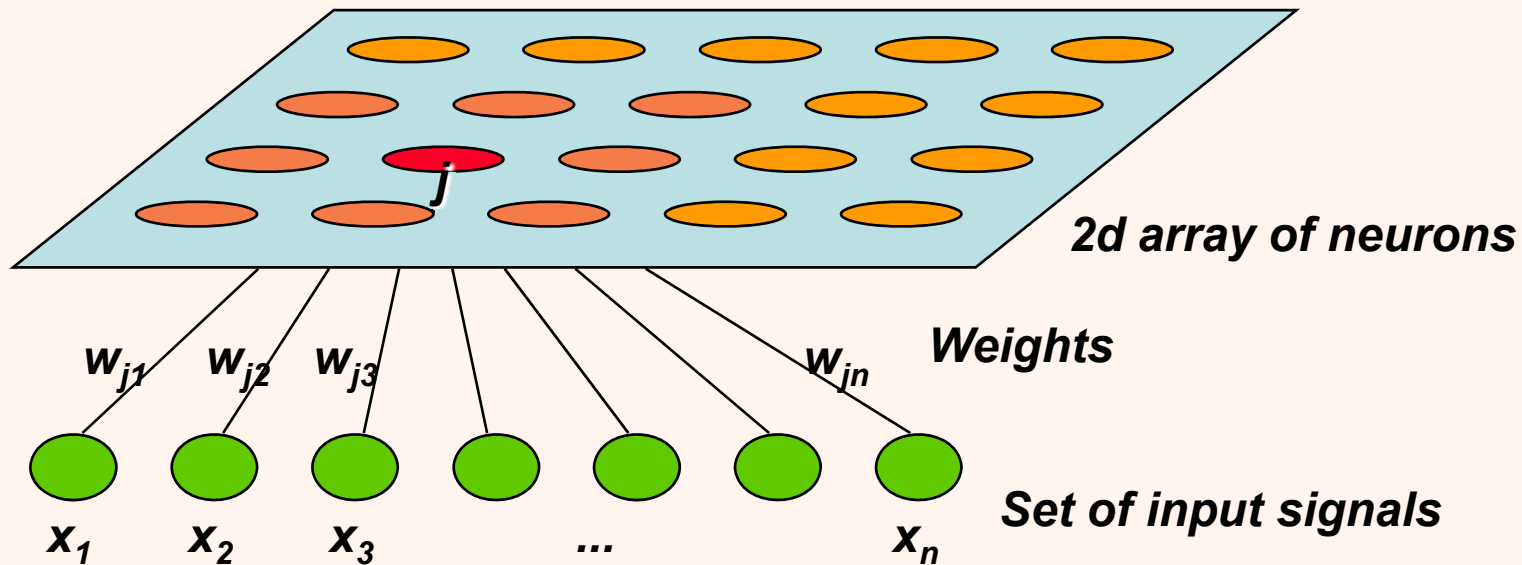
گام‌های مورد استفاده در SOM

- پس از انتخاب اولیه‌ی وزن‌ها سه گام صورت می‌پذیرد:
 - رقابت (competition)
 - همکاری (cooperation)
 - به‌روزرسانی وزن‌ها (synaptic adaptation)



توپولوژی دوبعدی

- بردار برنده از مشبک انتخاب می‌شود.
- نرون انتخاب شده به همراه همسایگانش با ضریب معینی فعال می‌گردد.
- هر چه به نرون برنده نزدیکتر باشیم میزان ضریب آموزش بالاتر خواهد بود.



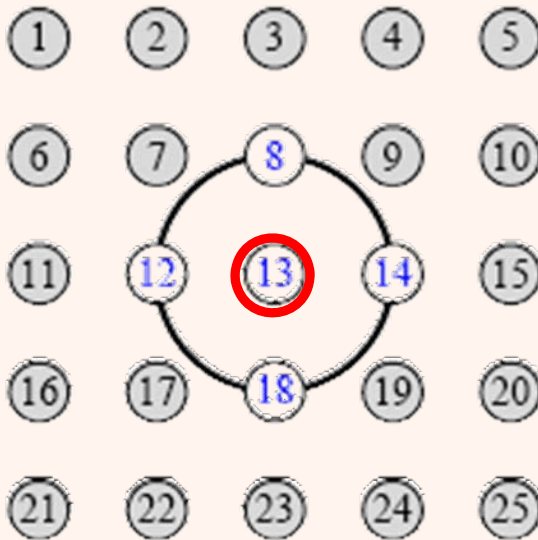
همسایگی

- اگر همسایگی را به وسیله عبارت زیر نشان دهیم، تمامی همسایه‌های یک نرون در فاصله‌ای کمتر یا مساوی d در نظر گرفته می‌شوند.

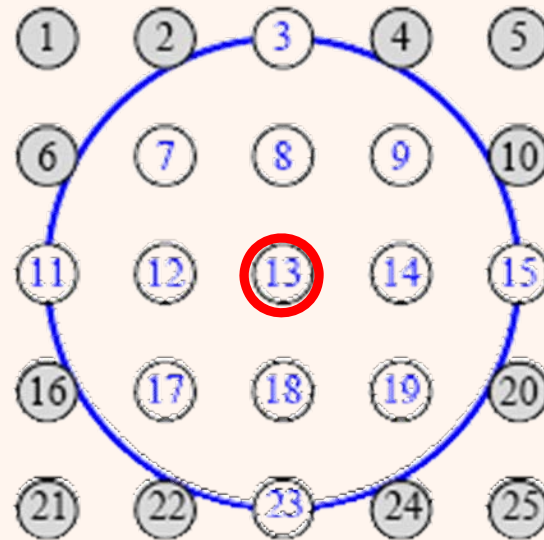
$$N_i(d) = \{j, d_{ij} \leq d\}$$

$$N_{13}(1) = \{8, 12, 13, 14, 18\} \text{ and}$$

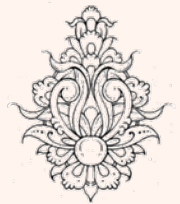
$$N_{13}(2) = \{3, 7, 8, 9, 11, 12, 13, 14, 15, 17, 18, 19, 23\}$$



$$N_{13}(1)$$



$$N_{13}(2)$$



SOM (SOFM)

- در چنین شبکه‌ای دو لایه بیشتر نداریم:
 - لایه‌ی ورودی
 - لایه‌ی خروجی
- هر واحد خروجی خود را با توجه به وزن‌هایی که بدان متصل است نشان می‌دهد.
 - می‌توان گفت شاخص هر خروجی در این صورت وزن‌هایش است.
- ابتدا وزن‌ها را به صورت تصادفی انتخاب می‌کنیم.



SOM (SOFM)

الگوریتم یادگیری

- وزن‌های شایفه‌ها به صورت تصادفی انتخاب می‌شوند.

تعداد گروه‌ها S

$$W_{i,j(n)} \quad 1 \leq i \leq R \quad 1 \leq j \leq S$$

وزن شایفه‌ی ورودی از A به J

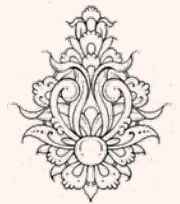
- برای هر واحد خروجی یک شعاع همسایگی N در نظر گرفته می‌شود.

- معیار فاصله را می‌توان ضرب داخلی دو بردار و یا فاصله‌ی اقلیدسی در نظر گرفت.

$$XW_j^T \quad \text{Max}$$

$$d_j = \sum_{i=1}^R (X_{i(n)} - W_{i,j(n)})^2 \quad \text{Min}$$

- واحد برنده انتخاب می‌شود: j^*



- به روز رسانی وزن‌های واحد برنده و همسایگانش

$$w_{ij(n+1)} = w_{ij(n)} + \eta [x_{i(n)} - w_{ij(n)}]$$

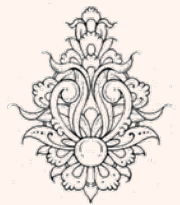
$$\text{for } j \in N_{j(n)}$$

شعاع همسایگی معمولاً در هر دوره متغیر است

- نرخ آموزش η نیز در این حالت متغیر است برای همسایه‌های نزدیک‌ترها به برنده، مقدار بیشتری دارد.

- و البته نرخ آموزش به تدریج کاهش می‌یابد.

$$0 < \eta(n) \leq \eta(n-1) \leq 1$$



SOM (SOFM)

تابع همسایگی

• یکی از مهمترین توابع همسایگی گاوسی است.

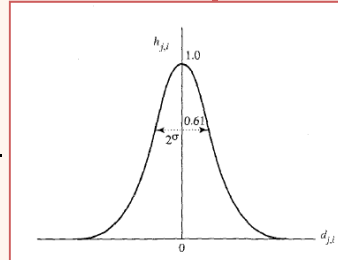
$$h_i(d_{i,j}) = \exp\left(-\frac{d_{i,j}^2}{2\sigma^2}\right)$$

– $d_{i,j}$ فاصله‌ی دو نرون را نشان می‌دهد اگر lattice یک بعدی باشد:

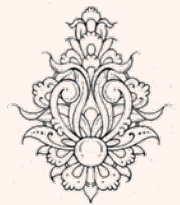
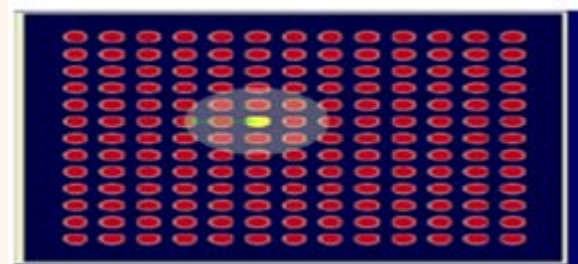
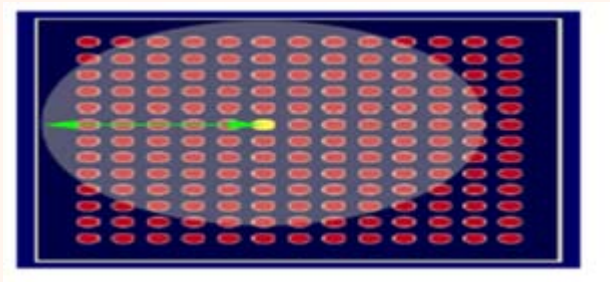
– $|z - i|$

– اگر lattice دو بعدی باشد:

– $\|r_j - r_i\|$



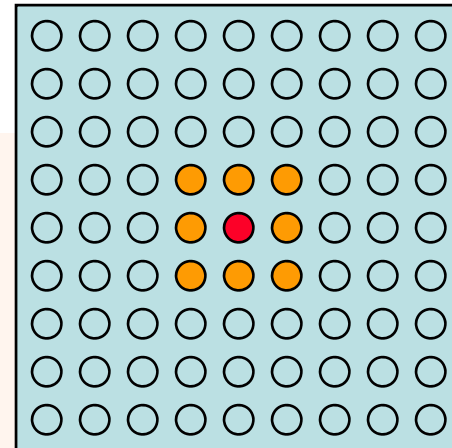
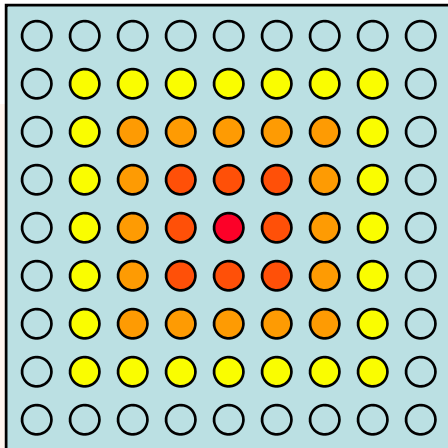
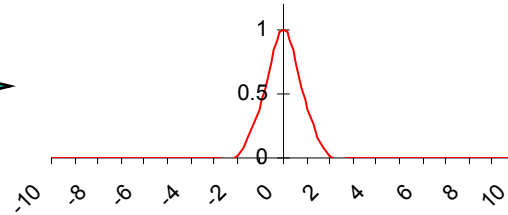
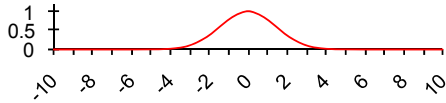
effective width



تابع همسایگی

- تابع همسایگی از شعاع همسایگی بزرگ شروع کند و به تدریج تمرکز تنها روی واحد برنده باقی می‌ماند.

$$h_i(d_{ij}) = \exp\left(-\frac{d_{ij}^2}{2\sigma^2}\right)$$



تابع همسایگی

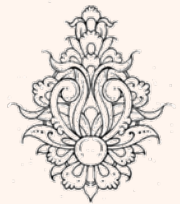
- تعداد همسایگان هم‌روند با زمان تغییر می‌کند.
- اگر نرون i برنده باشد d_{ij} همان فاصله‌ی نرون j از نرون برنده (i) است.

$$h_{j,i}(x) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2}\right)$$

- برای این منظور، انحراف معیار تابع گاوسی را متغیر در نظر گرفته می‌شود.

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \quad n=0,1,2,\dots$$

$$h_{j,i(x)}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right) \quad n=0,1,2,\dots$$



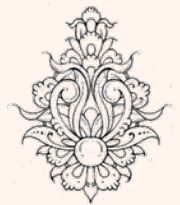
به روزسانی وزن‌ها

- بر طبق روابط به دست آمده خواهیم داشت:

$$w_j(n+1) = w_j(n) + \eta(n) h_{ij(x)}(n) (x - w_j(n))$$

- نرخ یادگیری را نیز به صورت نمایی متغیر در نظر گرفته می‌شود.

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right)$$



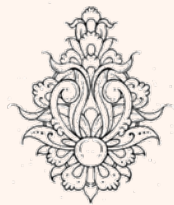
به روزسانی وزن‌ها (ادامه...)

$$w_j(n+1) = w_j(n) + \eta(n) h_{ij(x)}(n) (x - w_j(n))$$

$$h_{j,i(x)}(n) = \exp\left(-\frac{d_{j,i}^2}{2\sigma^2(n)}\right) \quad n = 0, 1, 2, \dots$$

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \quad n = 0, 1, 2, \dots$$

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right) \quad n = 0, 1, 2, \dots$$



به روزسانی وزن‌ها (ادامه...)

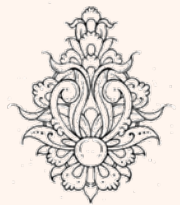
مقادیر η_0 ، σ_0 ، τ_0 و τ_1 به صورت تجربی در بازه‌های زیر خواهند بود:

$$\eta_0 = 0.1 \quad \tau_2 = 1000$$

این مقدار تدریجی کاهش می‌یابد اما نهایتاً بالاتر از ۰.۰۱ خواهد بود

- مقداری که برای σ_0 در نظر گرفته می‌شود برابر با شعاع مشبک است.
- بنابراین این مقدار وابسته به چیدمان نرون هاست.
- برای τ_1 مقدار زیر را در نظر می‌گیرند:

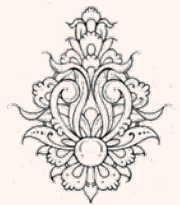
$$\tau_1 = \frac{1000}{\log \sigma_0}$$



- در SOM هر بردار که برنده باشد وزنهای خود و همسایگانش به نسبت همسایگی، تنظیم میشوند.

– هدف

- مکان جغرافیایی واحدها مشخص میشود.
- شعاع همسایگی به اندازهی تمامی واحدها شروع و سپس رو به کاهش میگذارد.
- تعداد تکرارها محدود ۱۰۰۰ در نظر گرفته میشود.
- در طی تکرارهای متوالی n کوچک و کوچکتر میشود.



Ordering وارد جزئیات نمیشود. به گونه ای فرم
زمن یا **Coarse** را نشان میدهد.

فازهای یادگیری (ادامه...)

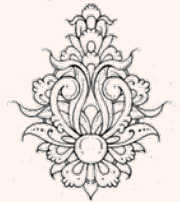
Convergence(tuning)

• هدف

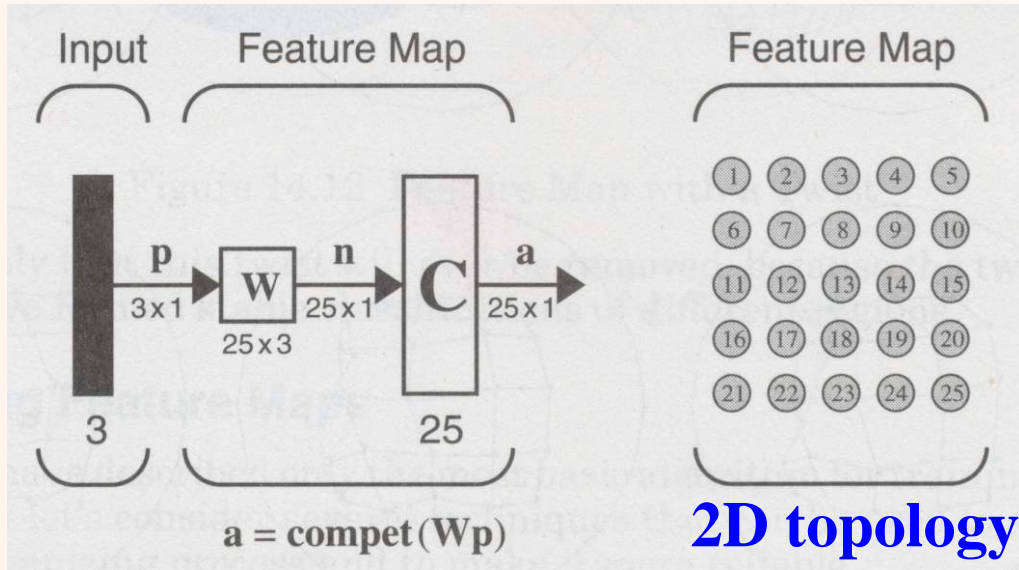
- جزییات تقسیم‌بندی گروه‌ها مشخص می‌شود.
- شعاع همسایگی به خود واحد برنده محدود خواهد شد.
- معمولاً تعداد تکرارها ۵۰۰ برابر تعداد واحدهای فروجی است.
 - تعداد گروه‌ها اهمیت دارد.
- نرخ آموزش کم و در حد ۰.۰۱ در نظر گرفته می‌شود.
- در این حالت جزییات هر واحد و نه همسایه‌ها اهمیت دارد.

• نکته:

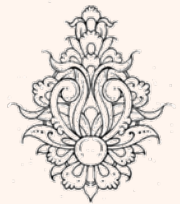
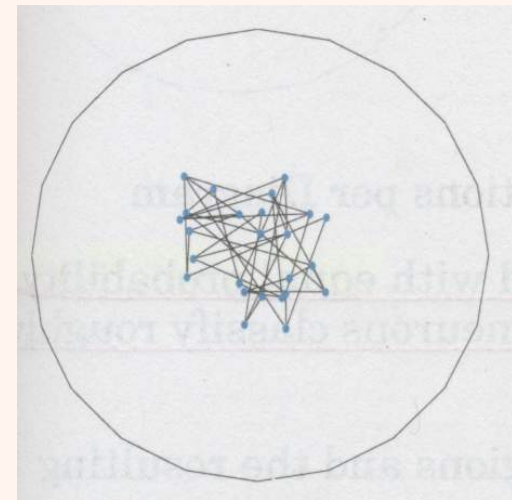
- بردارهای وزن اولیه مقادیر کوچکی انتخاب می‌شوند.
- اگر خیلی پراکنده باشند، مساله‌ی واحد مرده پیش می‌آید.



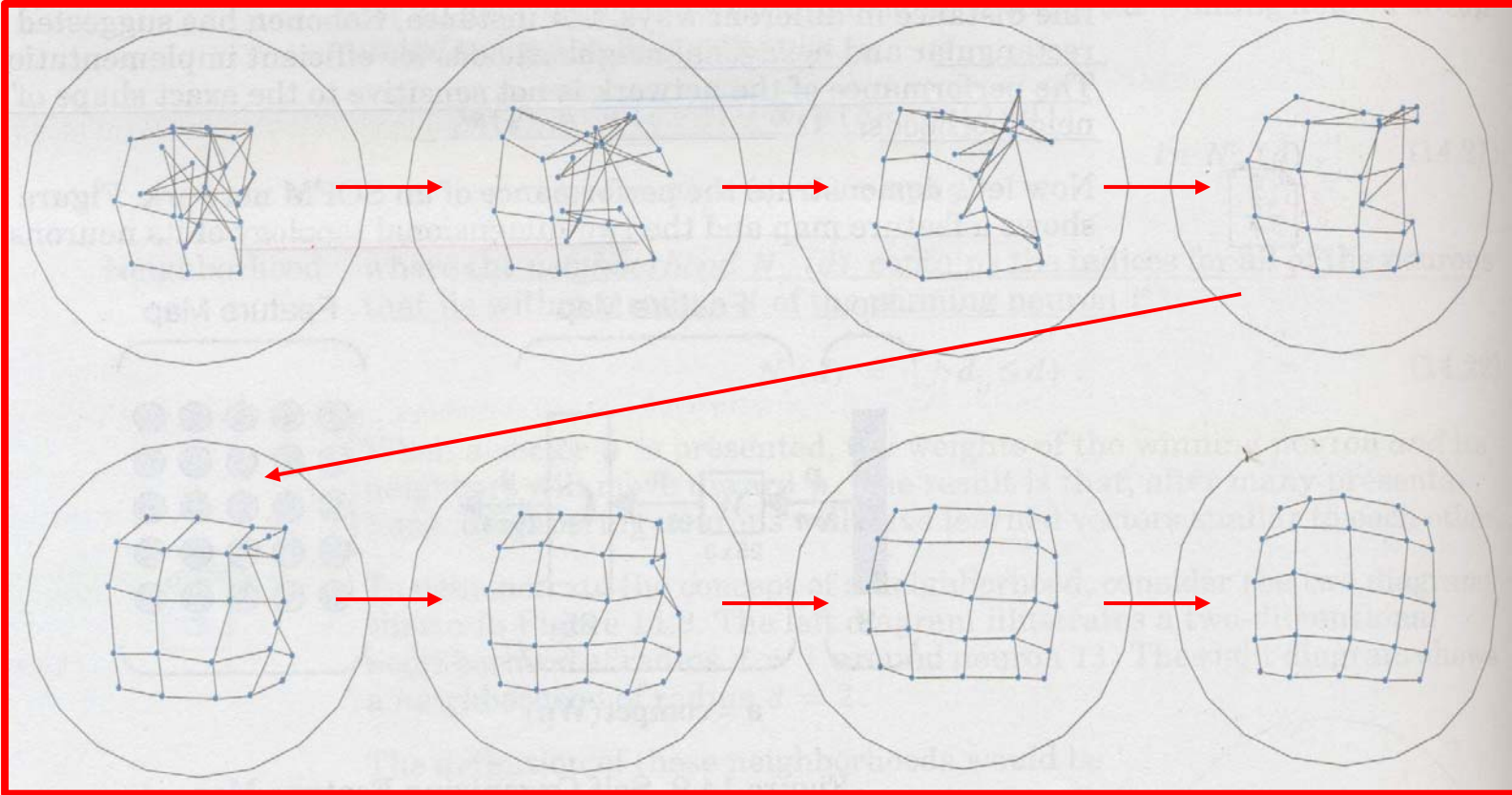
SOM (SOFM)



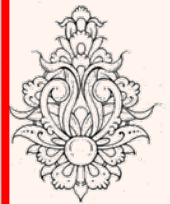
initial weight vectors



SOM (SOFM)



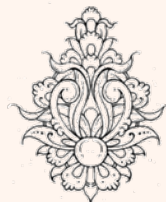
250 iterations per diagram



ژانسیکانه
سپهبدی
بهشتی

Renormalized SOM

- گفته شد که در ابتدا شعاع همسایگی بزرگ در نظر گرفته می‌شود. در این صورت با توجه به پارامترهای زیادی که درگیر هستند، آموزش کند خواهد شد.
- در این شیوه پیشنهاد می‌شود، شعاع همسایگی ثابت در نظر گرفته شود، اما نرون‌ها به تدریج در فرآیند آموزش دخیل شوند.



Luttrell, S. (1989). "Hierarchical vector quantisation (image compression)." *Communications, Speech and Vision, IEE Proceedings I* 136(6): 405-413.

جمع‌بندی SOM

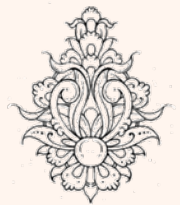
- گام‌های زیر می‌باید انجام شود:

1. مقداردهی اولیه (initialization)

- وزن‌ها به صورت تصادفی انتخاب می‌شوند، تنها محدودیت این است که وزن‌ها باید متفاوت باشند. مناسب‌تر مقدارهای انتخاب شده اندازه‌ی کوچکی داشته باشند.
- همچنین می‌توان از ورودی‌ها به عنوان وزن اولیه بهره جست.

2. نمونه‌برداری (sampling)

- یک الگوی ورودی به شبکه اعمال می‌شود.



جمع‌بندی SOM (ادامه...)

3. تطبیق شباهت (similarity matching)

- نرون برنده انتخاب شود.

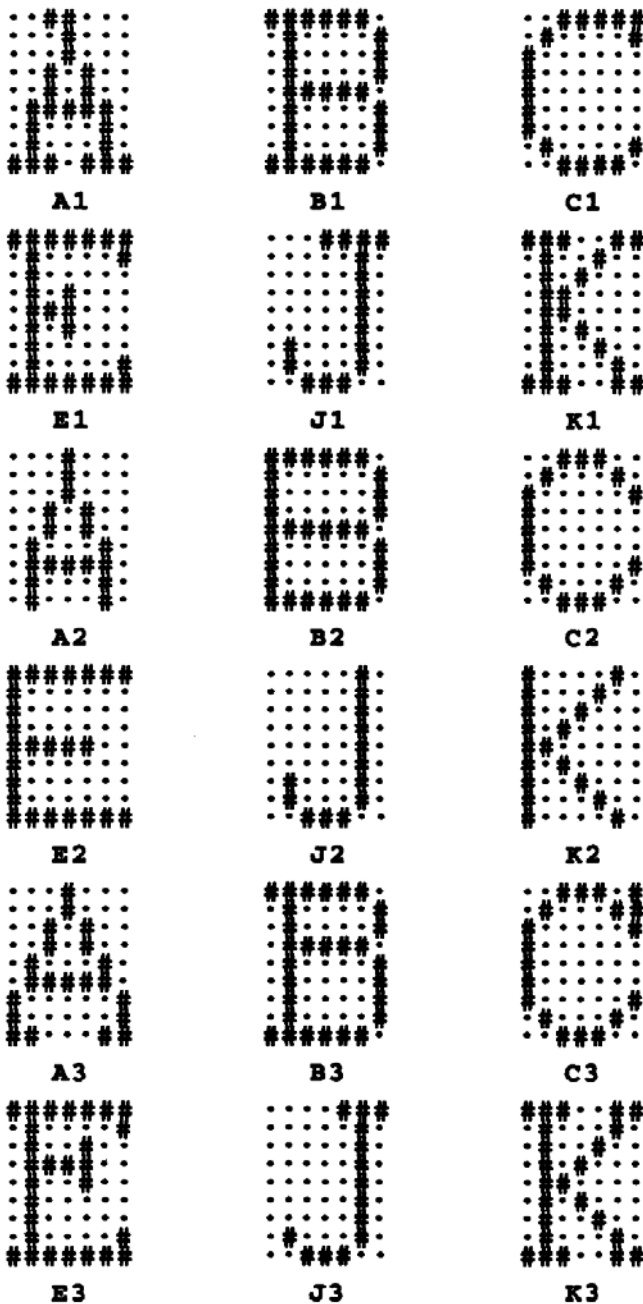
4. به‌روزرسانی (updating)

- طبق قانون آموزش مطرح شده، نرون برنده و همسایه‌هایش آموزش می‌بینند.

5. از گام دو مراحل از سر گرفته شوند، تا شبکه به پایداری برسد.

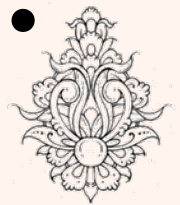


خوشه‌بندی کاراکتر



- ۲۱ الگوی ورودی و
- ۲۵ خوشه در نظر گرفته شده است.

- کاهش نرخ یادگیری به صورت خطی (از ۰.۶ تا ۰.۰۱)
- استفاده از سه توپولوژی مختلف



خوشه‌بندی بدون ساختار

no topological structure

| UNIT | PATTERNS |
|------|--------------------------------|
| 3 | C1, C2, C3 |
| 13 | B1, B3, D1, D3, E1, K1, K3, E3 |
| 16 | A1, A2, A3 |
| 18 | J1, J2, J3 |
| 24 | B2, D2, E2, K2 |

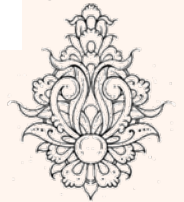
خوشه‌بندی اشتباه



linear structure (R=1)

| UNIT | PATTERNS | UNIT | PATTERNS |
|------|----------------|------|------------|
| 6 | K2 | 20 | C1, C2, C3 |
| 10 | J1, J2, J3 | 22 | D2 |
| 14 | E1, E3 | 23 | B2, E2 |
| 16 | K1, K3 | 25 | A1, A2, A3 |
| 18 | B1, B3, D1, D3 | | |

خوشبندی اشتباه

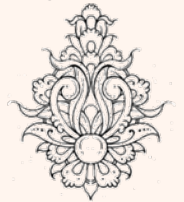


ساختار لوزی

diamond structure

| i\j | 1 | 2 | 3 | 4 | 5 |
|-----|------------|------------|------------|--------|--------|
| 1 | | J1, J2, J3 | | D2 | |
| 2 | C1, C2, C3 | | D1, D3 | | B2, E2 |
| 3 | | B1 | | K2 | |
| 4 | | | E1, E3, E3 | | A3 |
| 5 | | K1, K3 | | A1, A2 | |

خوشبندی اشتباه



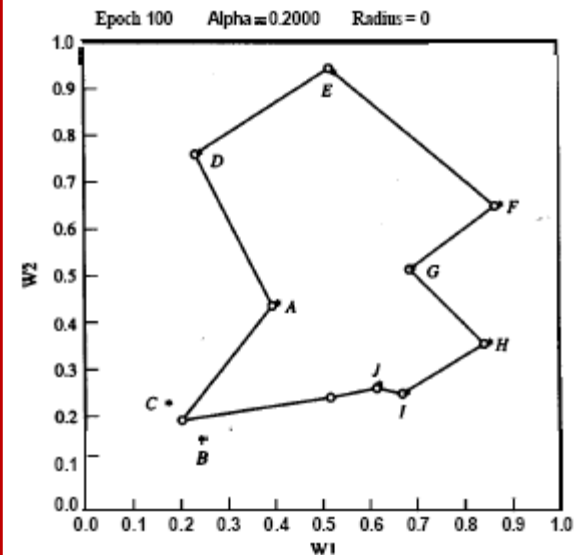
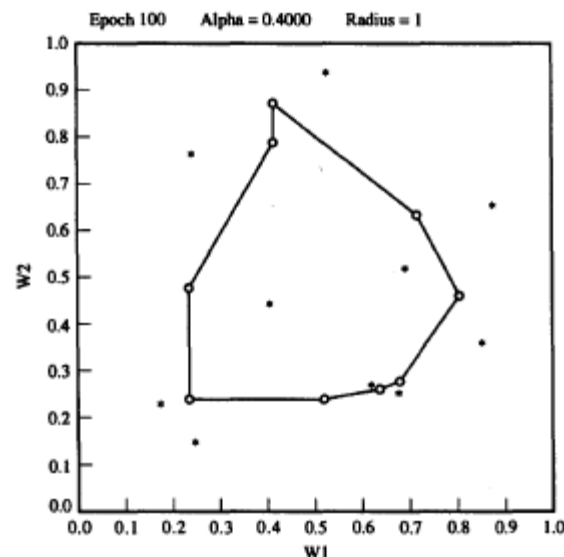
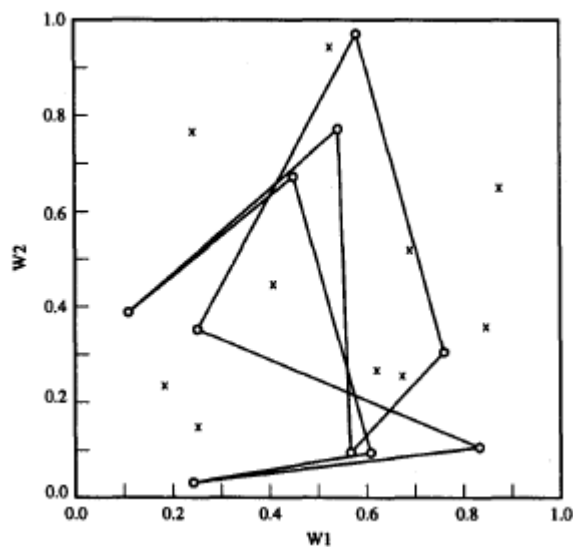
ORIGINAL CONTRIBUTION

Self-Organizing Feature Maps and the Travelling Salesman Problem

BERNARD ANGÉNIOL, GAËL DE LA CROIX VAUBOIS AND JEAN-YVES LE TEXIER

Thomson CSF/DSE

(Received March 1988; revised and accepted May 1988)



فروشندگی دوره‌گرد

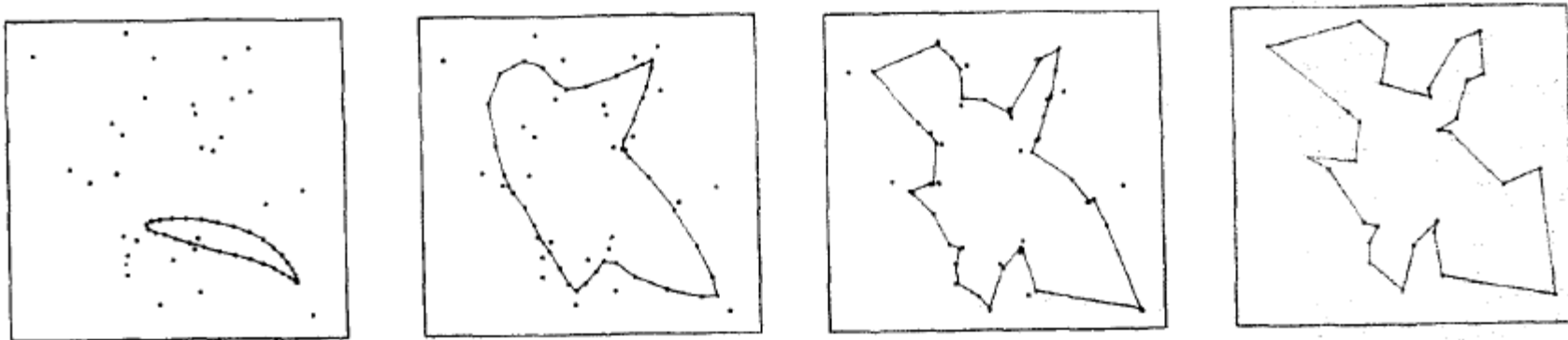


FIGURE 1. Evolution of the ring on a set of 30 cities used by Tank and Hopfield (1985). The solution shown here is the same as Lin-Kernighan's, and happens with probability 1.5×10^{-3} for $\alpha = 0.2$.

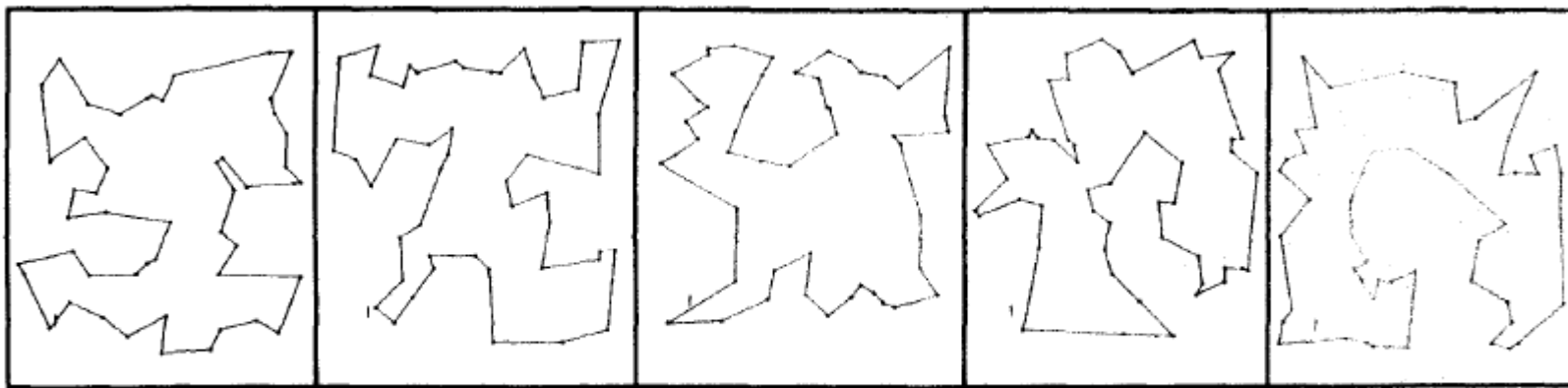


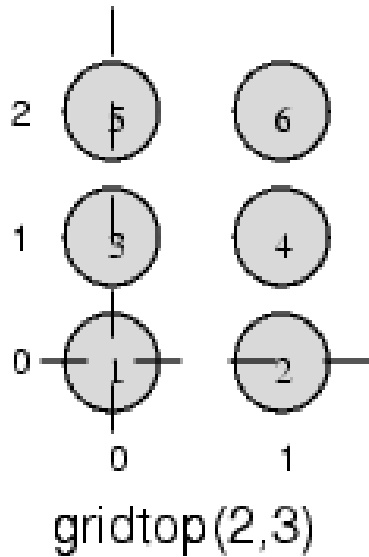
FIGURE 3. The best solution found on the five sets of 50 cities (Table 1c).



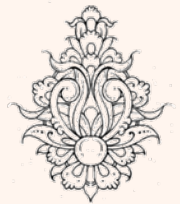
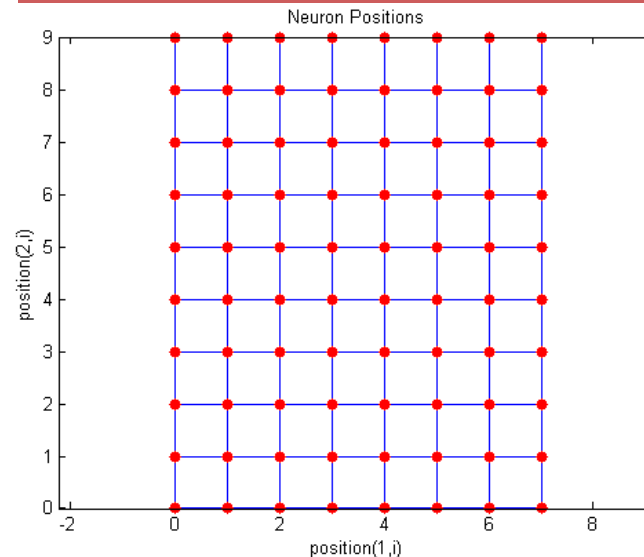
Matlab در SOM

- توپولوژی‌های مورد استفاده عبارتند از:
- مستطیلی (Gridtop)
- شش‌ضلعی (hextop)
- تصادفی (randtop)

```
pos = gridtop(2,3);
```

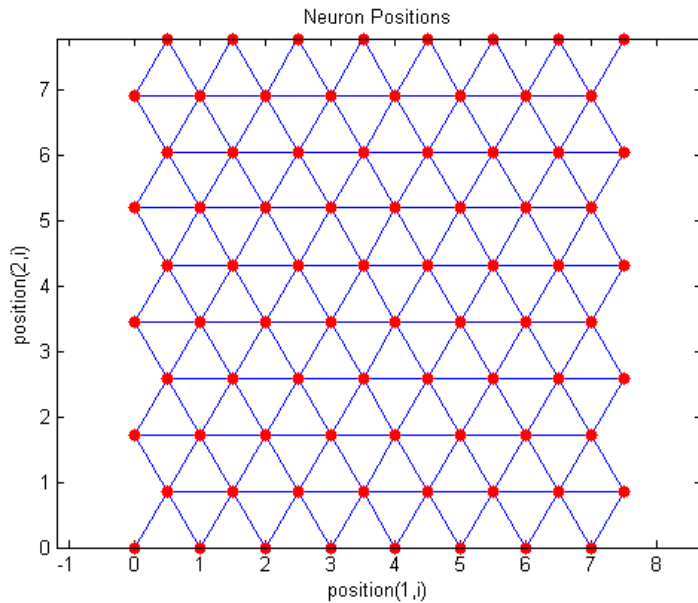


```
pos = gridtop(16,12);  
plotsom(pos);
```

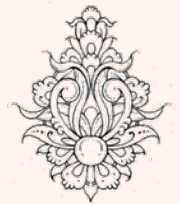
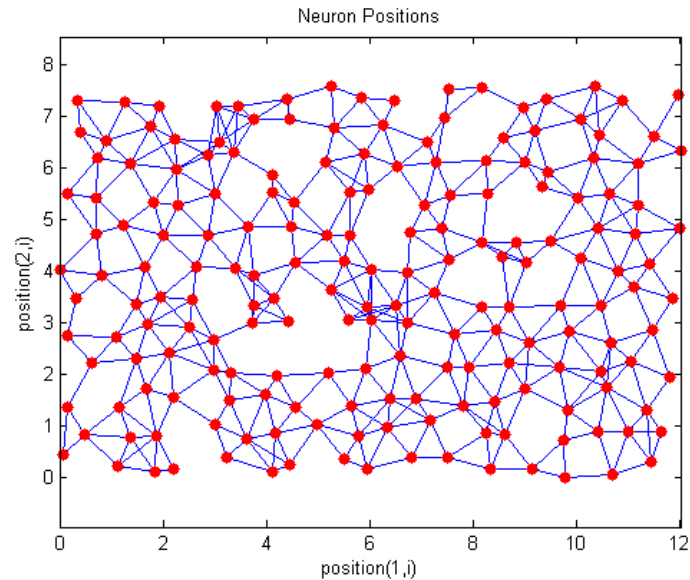


توپولوژیهای مورد استفاده

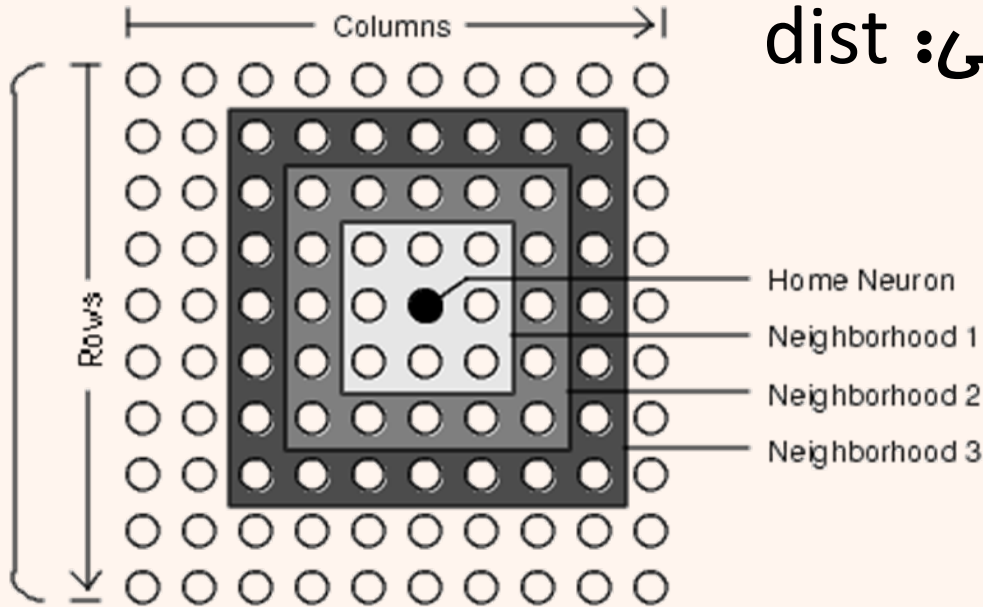
```
pos = hextop(8,10);  
plotsom(pos);
```



```
pos = randtop(16,12);  
plotsom(pos);
```



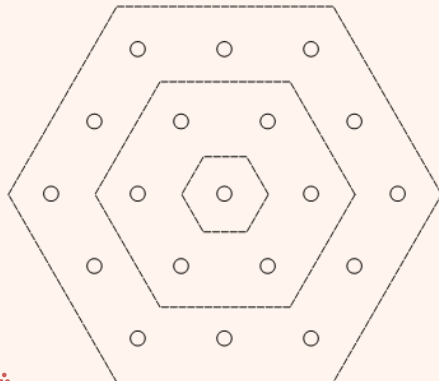
توابع فاصله



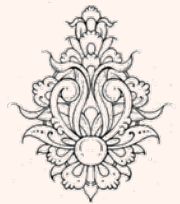
• فاصله ی اقلیدسی: dist

• Boxdist

• linkdist



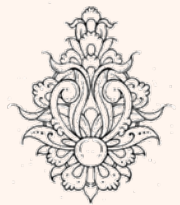
• $\text{sum}(\text{abs}(x-y))$: mandist

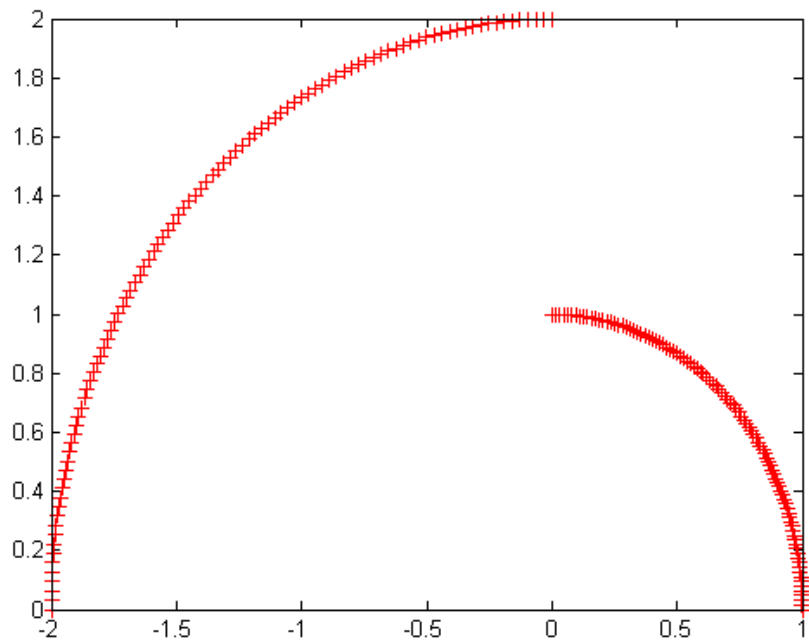


مثال

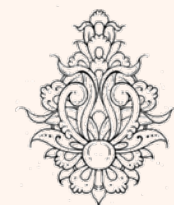
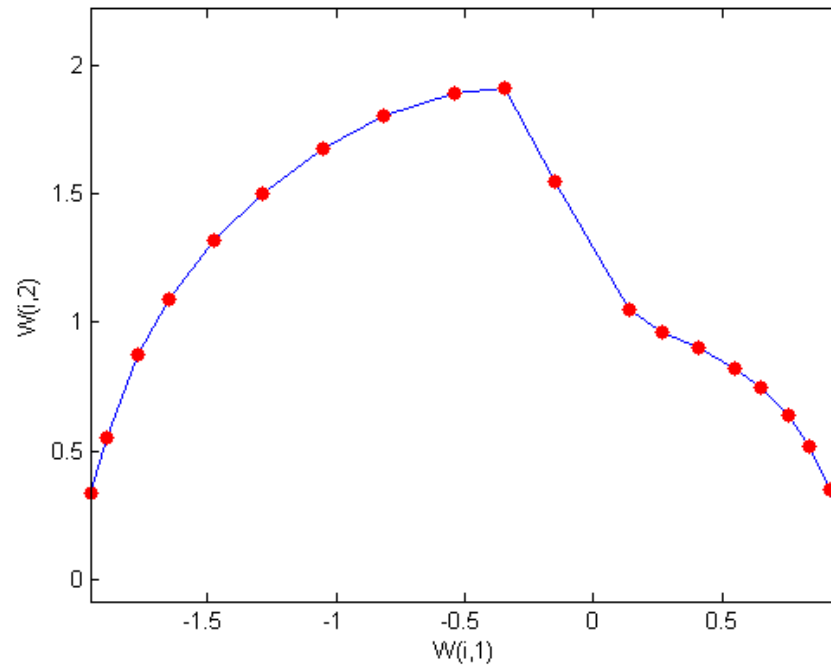
```
angles = 0:0.5*pi/99:0.5*pi;  
P1 = [sin(angles); cos(angles)];  
P2 = [-sin(angles); cos(angles)];  
P=[P1,2.*P2]  
plot(P(1,:),P(2,:),'+r')
```

```
net = newsom([0 1;0 1],[20]);  
net.trainParam.epochs = 10;  
net = train(net,P);  
figure;  
plotsom(net.iw{1,1},net.layers{1}.distances)  
p = [1;0];  
a = sim(net,p)
```



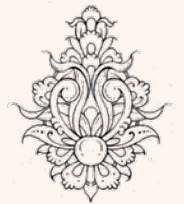
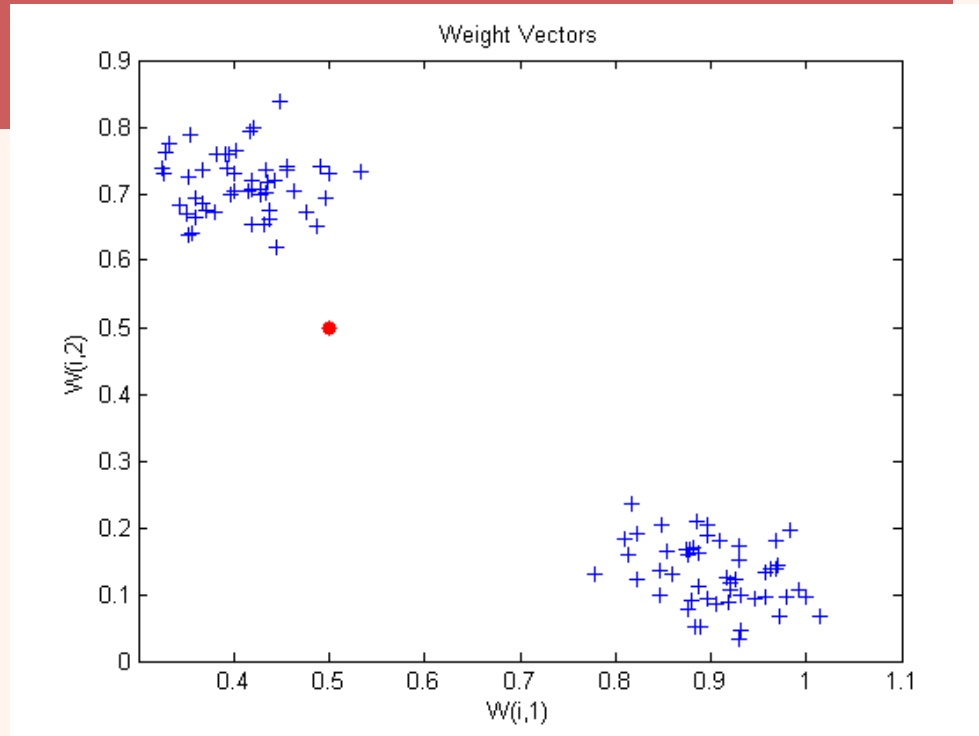


Weight Vectors

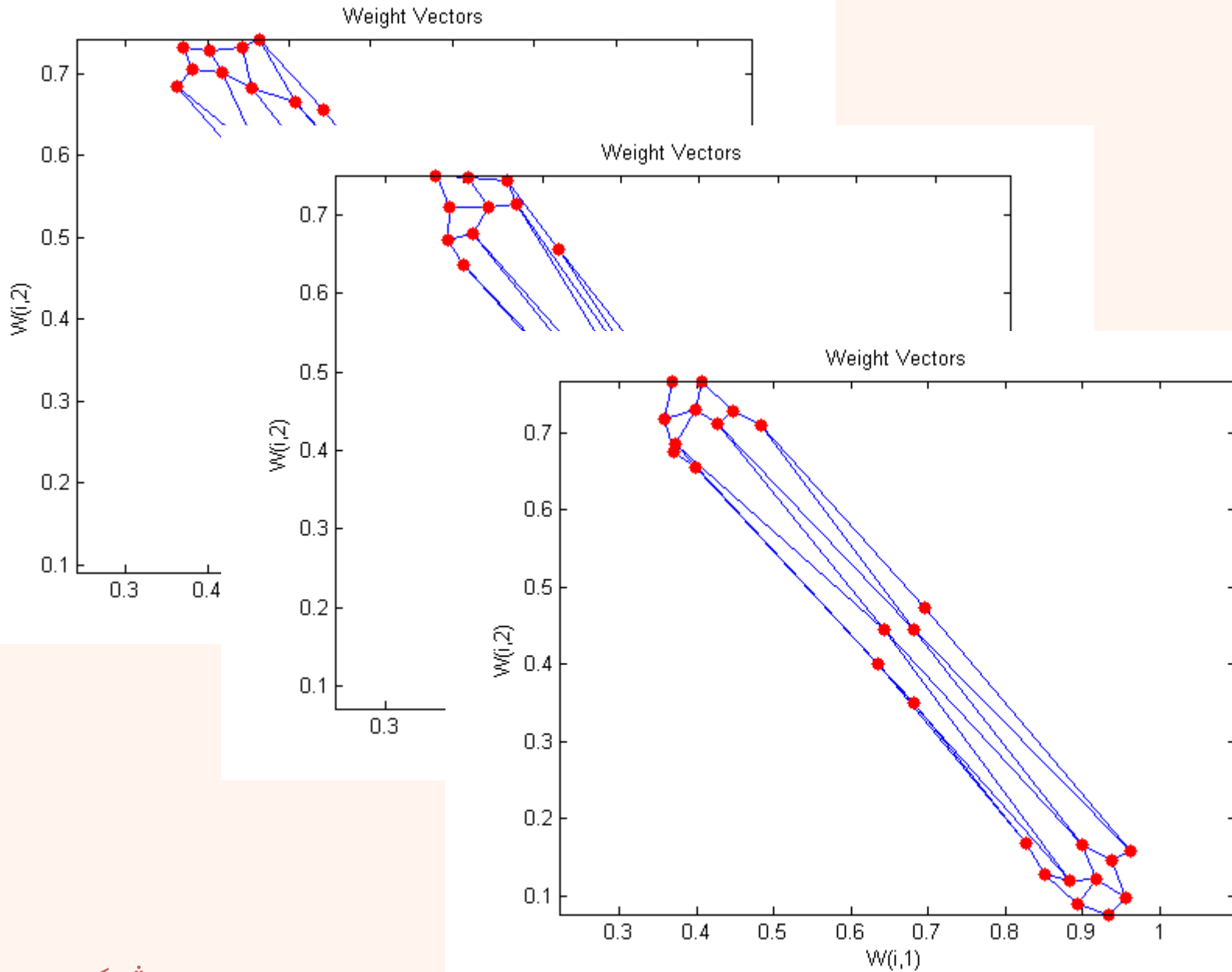


مثال

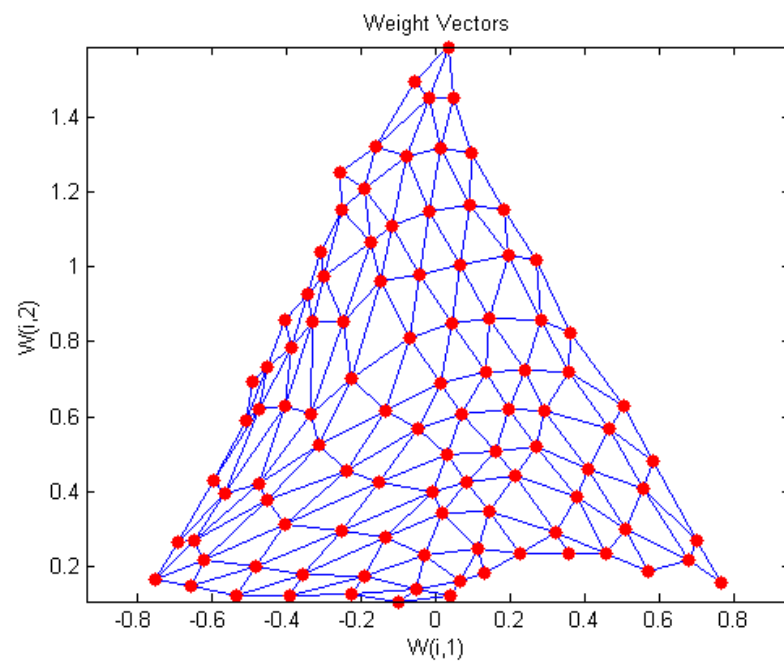
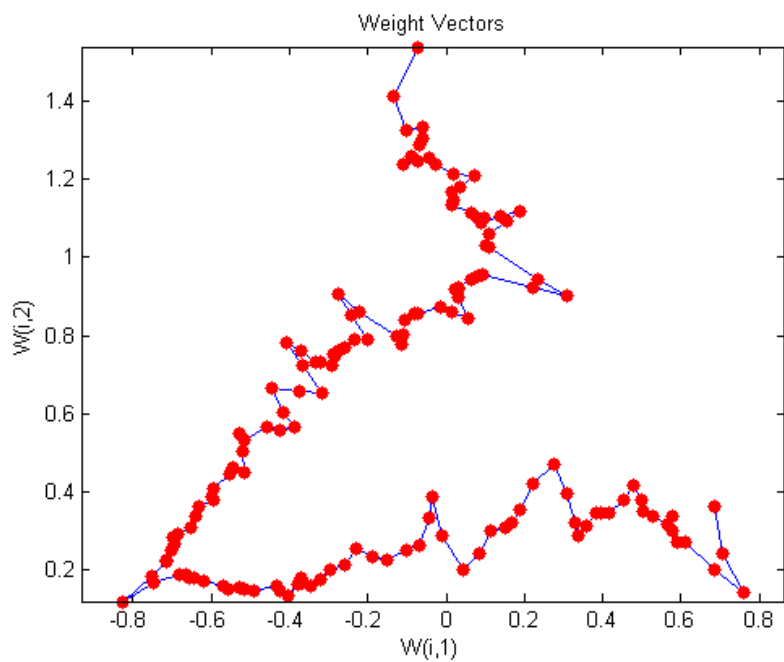
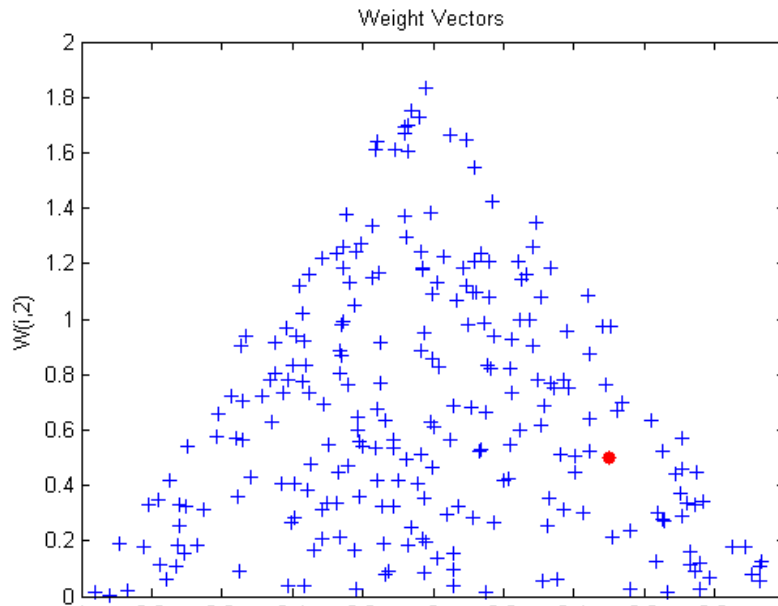
```
figure;  
plot(P(1,:),P(2,:),'+b')  
net = newsom([0 1; 0 1],[5 5],'gridtop');  
hold on;  
plotsom(net.iw{1,1},net.layers{1}.distances)  
net.trainParam.epochs = 10;  
net = train(net,P);  
figure;  
plotsom(net.iw{1,1},net.layers{1}.distances)  
p = [0.5;0.3];  
a = sim(net,p)
```

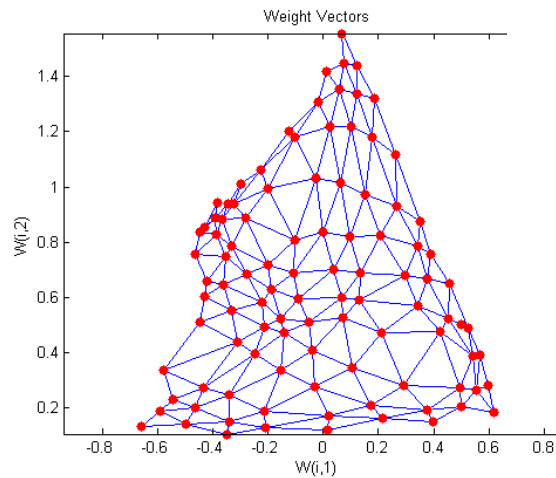
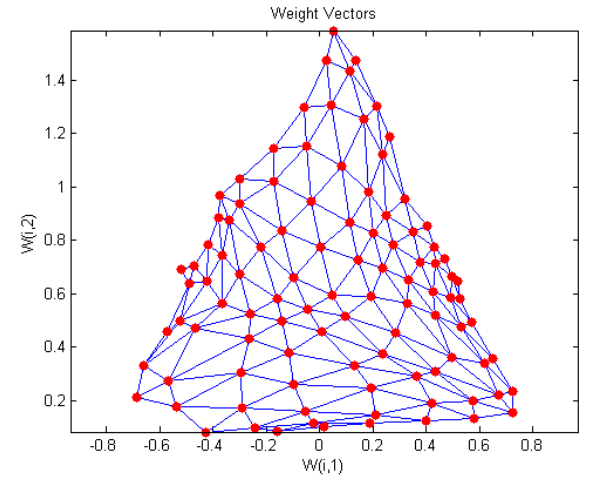
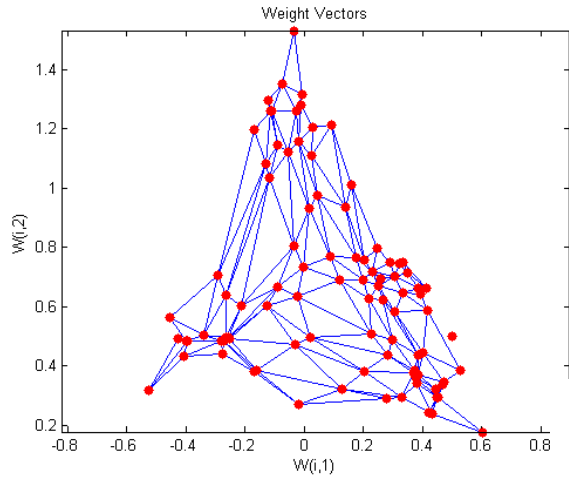
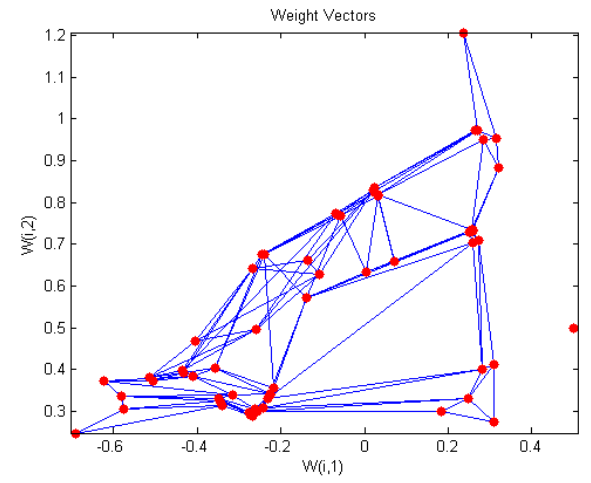
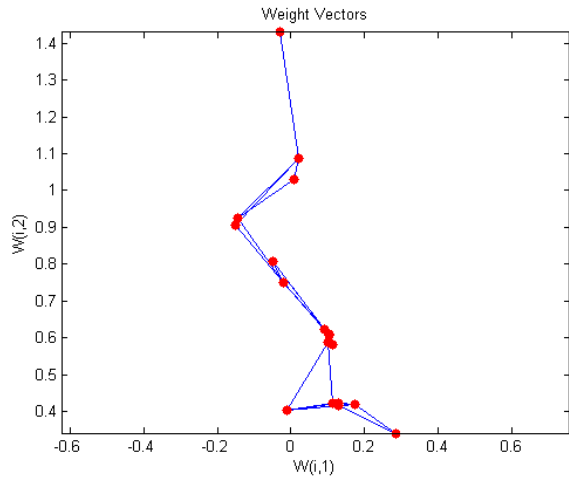


ادامه‌ی مثال



مثال





خصوصیات SOM

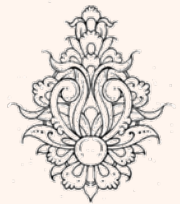
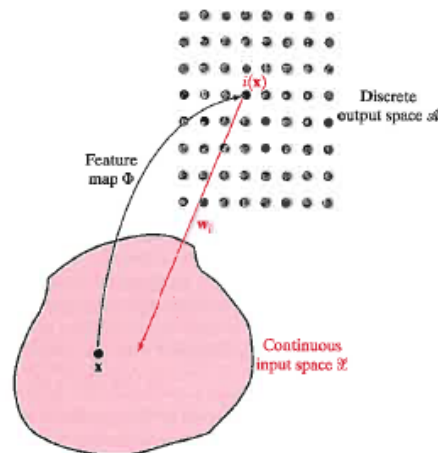
- پس از این آموزش به پایان برسد، وزن‌های به دست آمده به نوعی حاوی خواص آماری داده‌های ورودی خواهند بود.

- به نوعی چگالی توزیع را نشان می‌دهد.

- برای انتخاب بردار خصیصه قابل استفاده خواهد بود.

$$\Phi : H \rightarrow A$$

FIGURE 9.4 Illustration of the relationship between feature map Φ and weight vector w_i of winning neuron i .

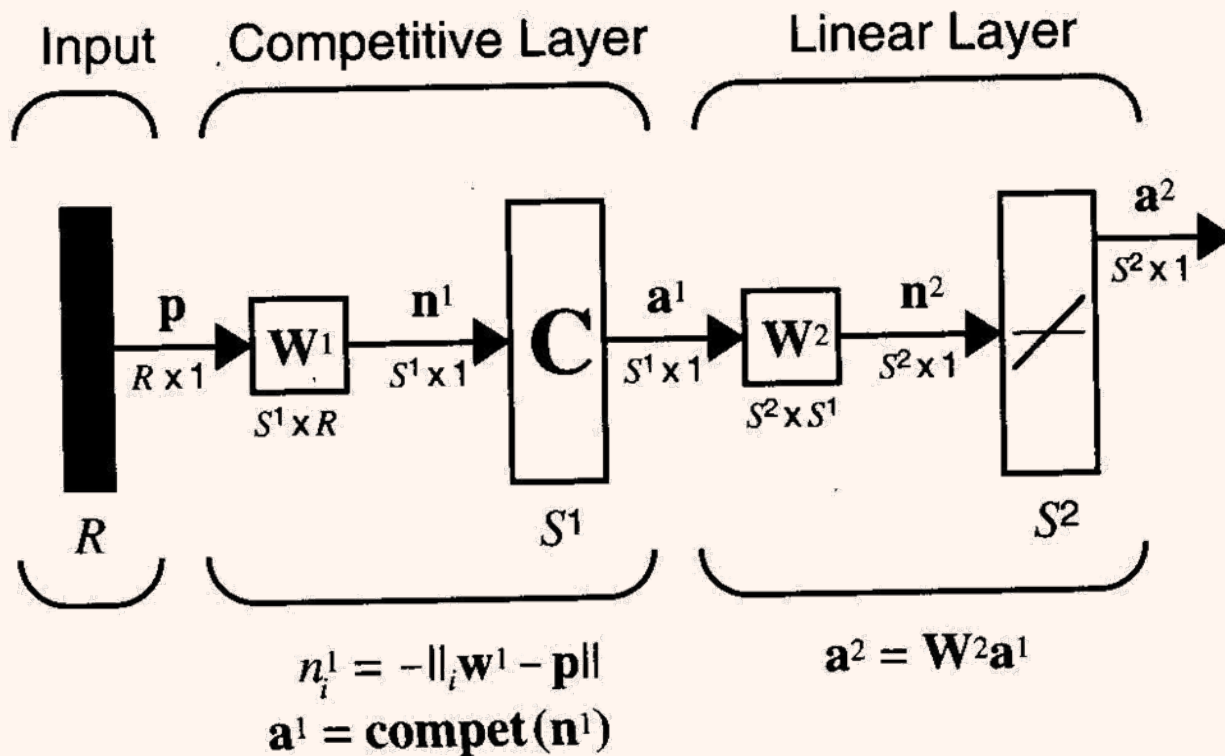


LVO

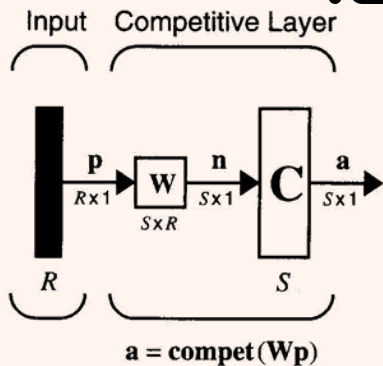


Learning Vector Quantization

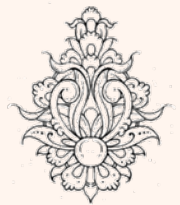
این شبکه یک شبکه ترکیبی است که به هر دو شیوهی supervised و unsupervised آموزش می‌بیند.



- در نخستین لایه، هر **کلاس** به **یک** یا **چند نرون** واگذار می‌شود.
- تعداد نرون‌ها در این لایه باید حداقل به تعداد کلاس‌ها باشد.
- در لایه‌ی دوم، به ازای هر کلاس، **یک نرون** وجود دارد.
- تعداد نرون‌ها با تعداد کلاس‌ها برابرست.

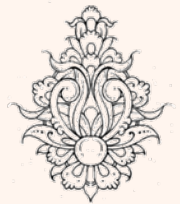
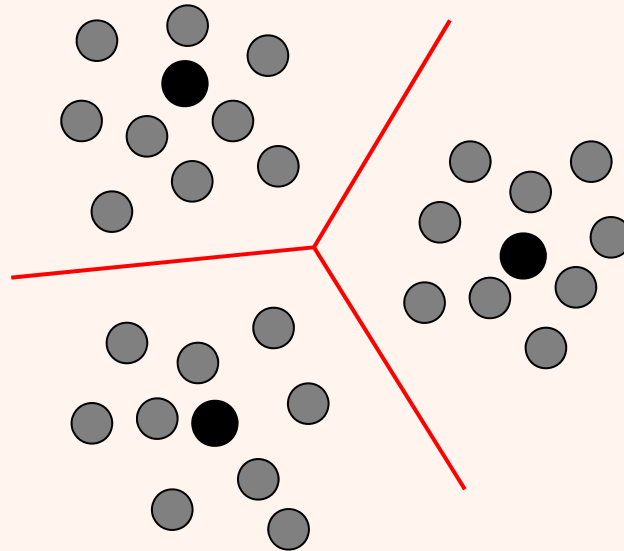


$$n_i^1 = -\|_i \mathbf{w}^1 - \mathbf{p}\| \Rightarrow \mathbf{n}^1 = - \begin{bmatrix} \|_1 \mathbf{w}^1 - \mathbf{p}\| \\ \|_2 \mathbf{w}^1 - \mathbf{p}\| \\ \vdots \\ \|_{S^1} \mathbf{w}^1 - \mathbf{p}\| \end{bmatrix} \Rightarrow \mathbf{a}^1 = \text{compet}(\mathbf{n}^1)$$



زیر کلاس‌ها

- بدین ترتیب در لایه‌ی اول هر کلاس به یک سری «زیرکلاس» تقسیم می‌شود.
- برنده‌ی لایه‌ی اول مشخص می‌کند که فروجی به کدام زیرکلاس تعلق دارد.

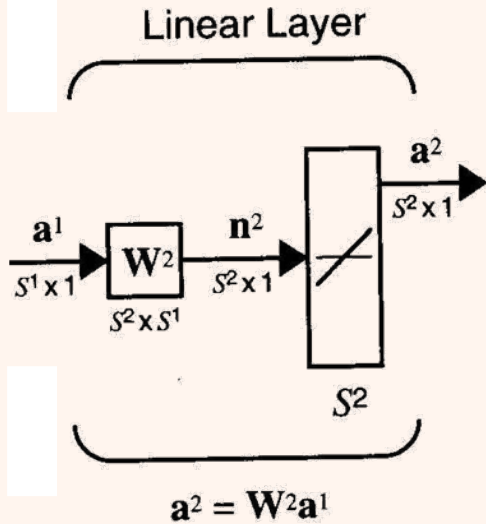


زیر کلاس‌ها (ادامه...)

- در لایه‌ی بعدی این که زیر کلاس‌ها برنده شده به کدام کلاس تعلق دارد، بررسی می‌شود.
- بدین ترتیب و برخلاف شبکه‌ی رقابتی در این حالت مزرهای کلاس‌ها پیچیده‌تر خواهند شد و قابلیت جداسازی کلاس‌های غیرمحدب را خواهد داشت.



لایه‌ی خطی



- سطرهای ماتریس بیانگر کلاس‌ها و ستون‌ها بیانگر زیرکلاس‌ها هستند.

- در هر ستون فقط یک عنصر یک خواهیم داشت.

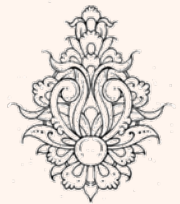
$w_{ki}^2 = 1 \Rightarrow$ زیر کلاس i بخشی از کلاس k است.

$$W^2 = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Subclasses 1 & 3 belong to class 1

Subclasses 2 & 5 belong to class 2

Subclass 4 belongs to class 3



شیوهی آموزش

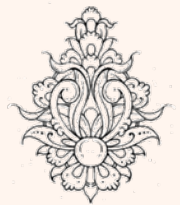
- آموزش رقابتی با ناظر همراه می‌شود.
- وزن‌های W^2 مشخص می‌شوند، معمولاً به هر کلاس تعداد زیر کلاس یکسانی نسبت داده می‌شود.
- وزن‌های لایه‌ی اول به صورت زیر آموزش می‌بینند:
 - اگر نرون لایه‌ی اول به درستی برنده شده باشد:
 - به سمت بردار ورودی (p) حرکت داده می‌شود.

$$w_{i^*}^1(q) = w_{i^*}^1(q-1) + \alpha(p(q) - w_{i^*}^1(q-1)) \quad \text{if } a_{k^*}^2 = t_{k^*} = 1$$

– وگرنه:

- از بردار ورودی دور می‌شود.

$$w_{i^*}^1(q) = w_{i^*}^1(q-1) - \alpha(p(q) - w_{i^*}^1(q-1)) \quad \text{if } a_{k^*}^2 \neq t_{k^*}$$



شیوهی آموزش (ادامه...)

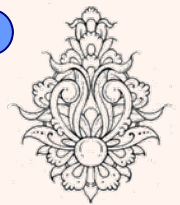
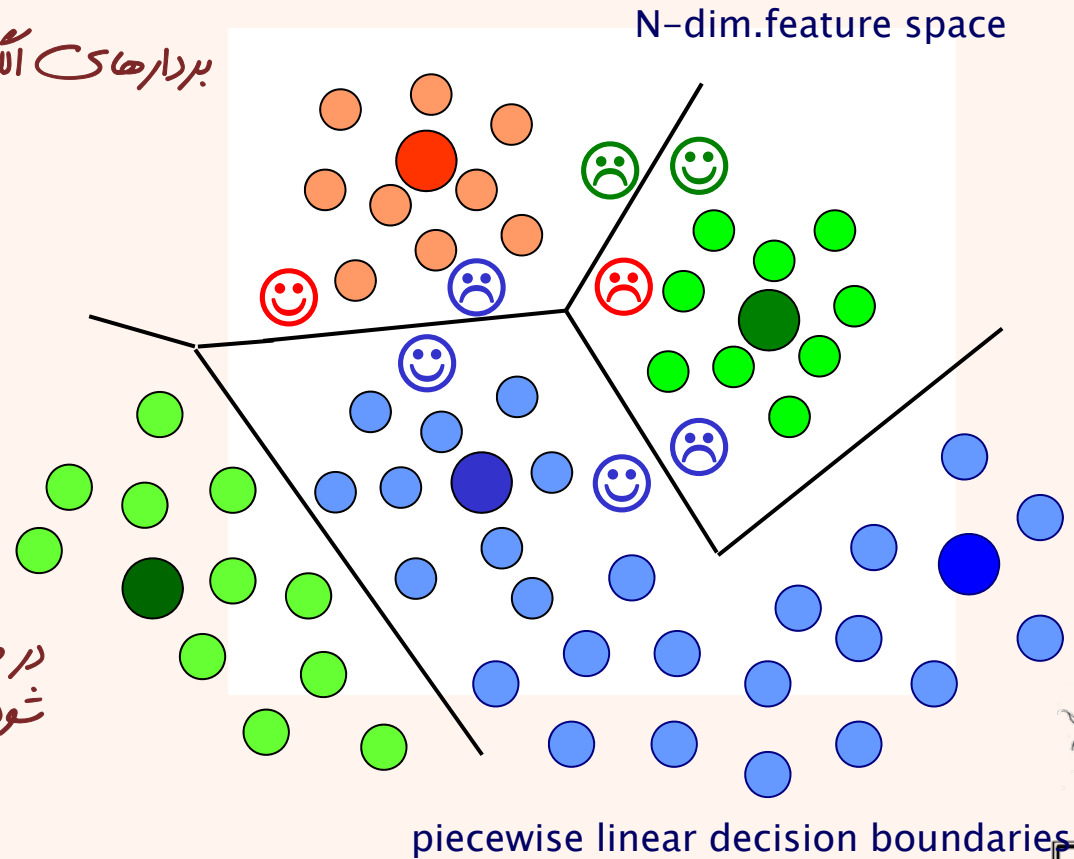
بردارهای آلو مقداردهی اولیه می‌شوند

یک ورودی اعمال می‌شود

نزدیک‌ترین بردار آلو انتخاب شود

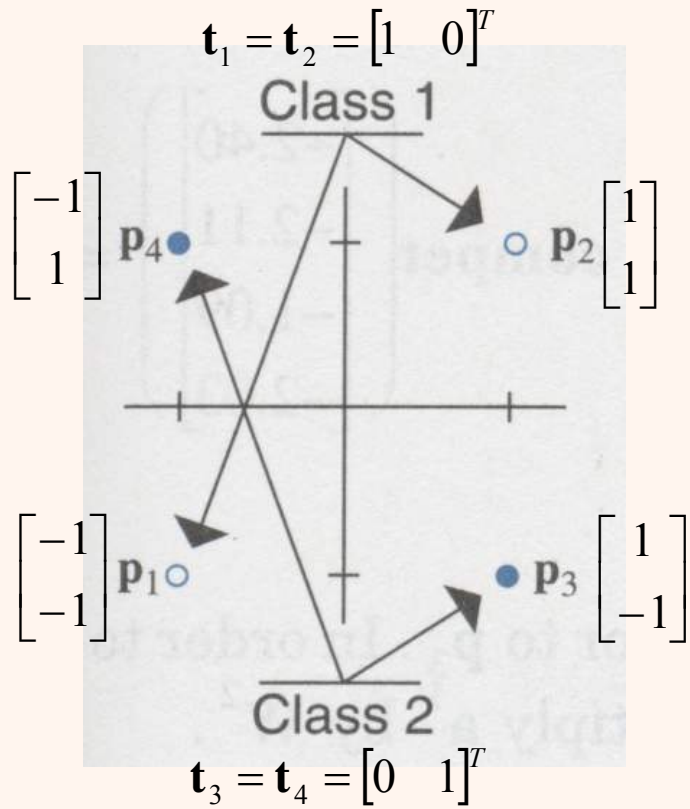
در صورت درست بودن آلو به بردار ورودی نزدیک شود

در صورتی که درست نیست از ورودی دور شود

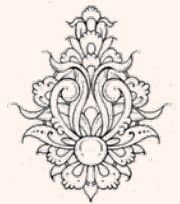
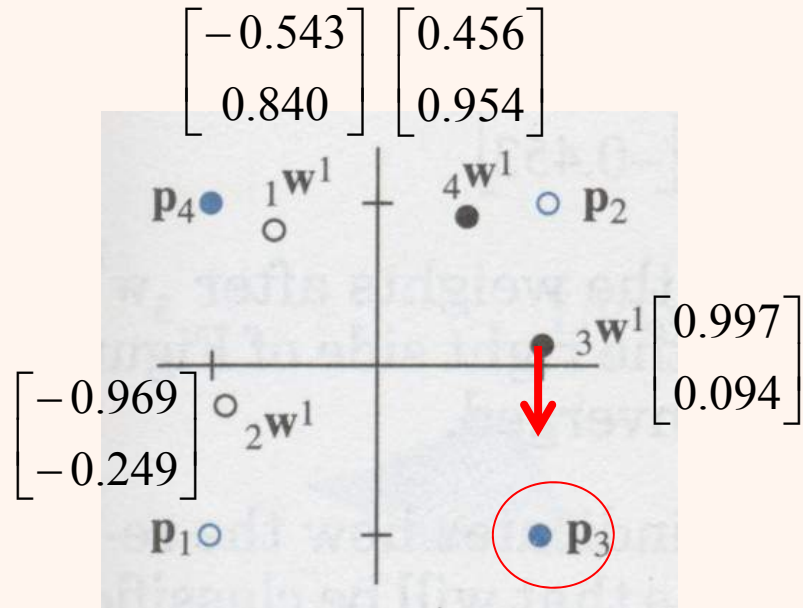


$$w^*(t+1) = w^*(t) \pm \eta_w (\xi^{(t)} - w^*(t))$$

مثال



$$\mathbf{W}^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

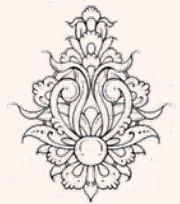


مثال-فرآیند آموزش

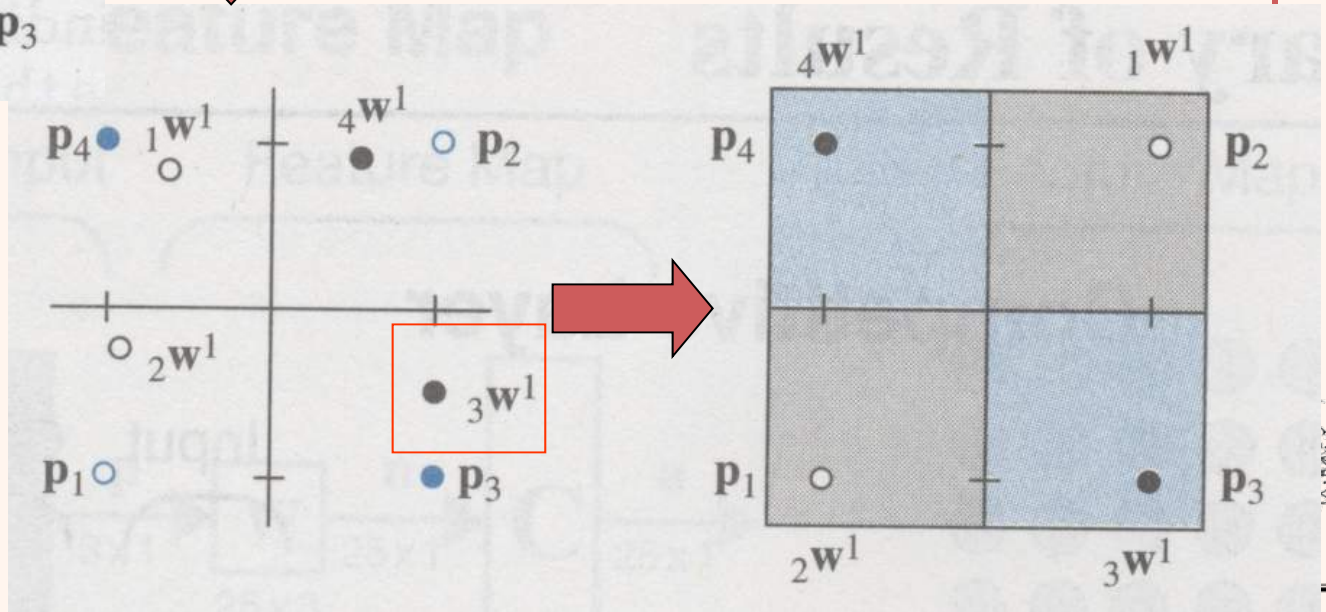
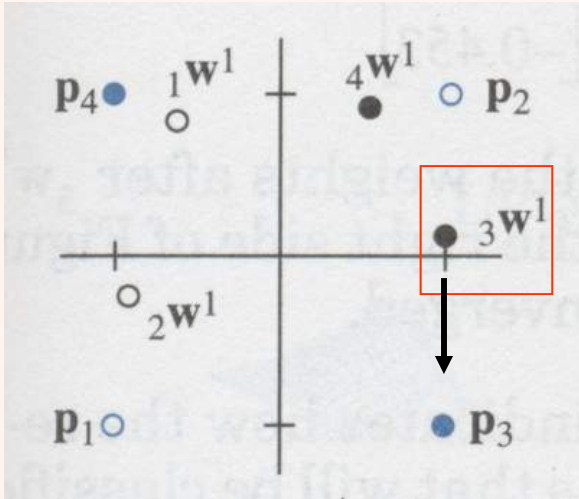
$$\mathbf{a}^1 = \text{compet}(\mathbf{n}^1) = \text{compet} \left(- \begin{bmatrix} \left\| \mathbf{w}^1 - \mathbf{p} \right\|_1 \\ \left\| \mathbf{w}^1 - \mathbf{p} \right\|_2 \\ \left\| \mathbf{w}^1 - \mathbf{p} \right\|_3 \\ \left\| \mathbf{w}^1 - \mathbf{p} \right\|_4 \end{bmatrix} \right) = \text{compet} \left(\begin{bmatrix} -2.40 \\ -2.11 \\ -1.09 \\ -2.03 \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$\mathbf{a}^2 = \mathbf{W}^2 \mathbf{a}^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \mathbf{t}_3$$

$${}_3 \mathbf{w}^1(1) = {}_3 \mathbf{w}^1(0) + 0.5(\mathbf{p}_3 - {}_3 \mathbf{w}^1(0)) = \begin{bmatrix} 0.998 \\ -0.453 \end{bmatrix}$$



مثال-فرآیند آموزش



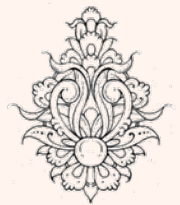
ژانسه کانه
سپهر سید
بهشتی

برخی مشکلات و موانع

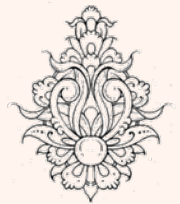
- با این که LVQ در موارد بسیاری از عهده‌ی حل مسائل بر می‌آید، با محدودیت‌هایی مواجه خواهد بود:

- ممکن است در این حالت نیز «**نرون مرده**» داشته باشیم، در این جا نیز می‌توان از «**مکانیزه وجدان**» استفاده نمود.

- در برخی موارد لازم است یک نرون به نامیه‌ای دیگر حرکت کند و در این بین ممکن است مجبور به گذر از نامیه‌ای باشد که به کلاس او تعلق ندارد. با توجه به دفع توسط آن کلاس امکان عبور نخواهد یافت و در نتیجه به درستی آموزش نخواهد دید.



- برای رفع این مشکل قانون آموزش kohonen به شیوهی زیر بهبود می‌باید:
- در صورتی که کلاس‌بندی به درستی انجام شده باشد، آموزش مانند قبل انجام می‌شود.
- و گرنه، زیرکلاس انتخاب شده از نمونه‌ی آموزشی دور شده و نزدیک‌ترین کلاس به نمونه‌ی آموزشی به آن نزدیک خواهد شد. الگوریتم حاصل LVQ2 نام دارد.



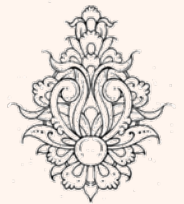
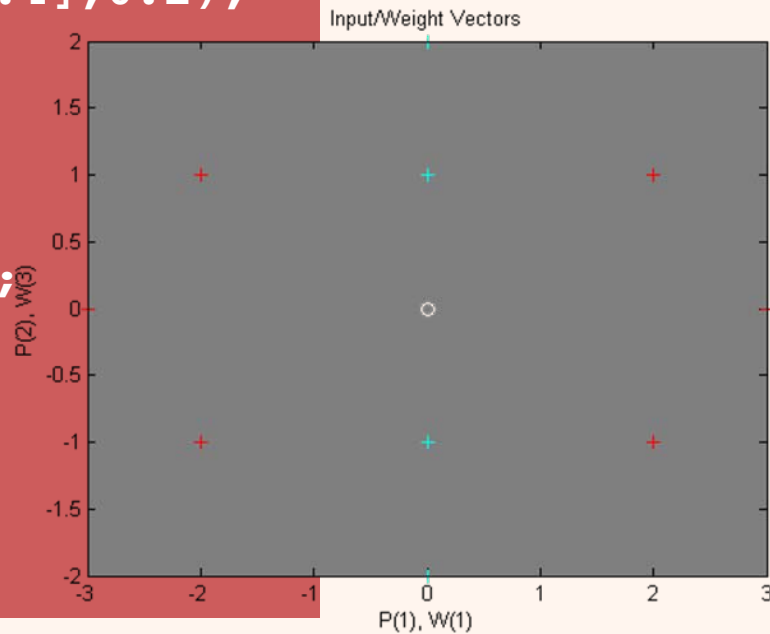
مثال

```
P = [-3 -2 -2 0 0 0 0 +2 +2 +3;  
      0 +1 -1 +2 +1 -1 -2 +1 -1 0];  
C = [1 1 1 2 2 2 2 1 1 1];  
T = ind2vec(C);
```

```
colormap(hsv);  
plotvec(P,C)  
title('Input Vectors');  
xlabel('P(1)');  
ylabel('P(2)');
```

```
net = newlvq(minmax(P),4,[.6 .4],0.1);
```

```
hold on  
W1 = net.IW{1};  
plot(W1(1,1),W1(1,2),'ow')  
title('Input/Weight Vectors');  
xlabel('P(1), W(1)');  
ylabel('P(2), W(3)');
```



ادامه‌ی

```
net.trainParam.epochs=150;  
net=train(net,P,T);
```

```
cla;  
plotvec(P,C);  
hold on;  
plotvec(net.IW{1}',vec2ind(net.LW{2}),'o');  
W1 = net.IW{1};  
hold on;  
voronoi(W1(:,1),W1(:,2))
```

```
p = [0.2; 1];  
a = vec2ind(sim(net,p))
```

