شبکه‌های عصبی مصنوعی

درس ۳

یک مثال گویا از شبکه‌ی عصبی

An Illustrative Example

کاظم فولادی قلعه
دانشکده مهندسی، پردیس فارابی
دانشگاه تهران

http://courses.fouladi.ir/nn

An Illustrative Example

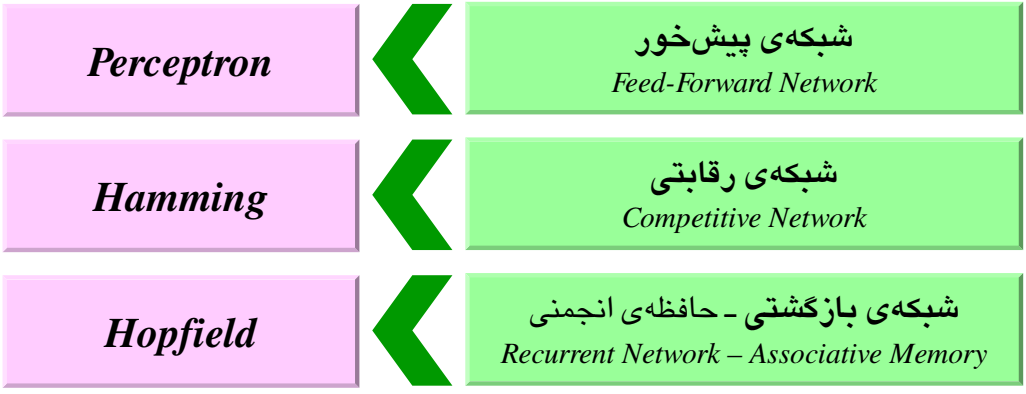
یک مثال گویا از شبکه‌ی عصبی

۱

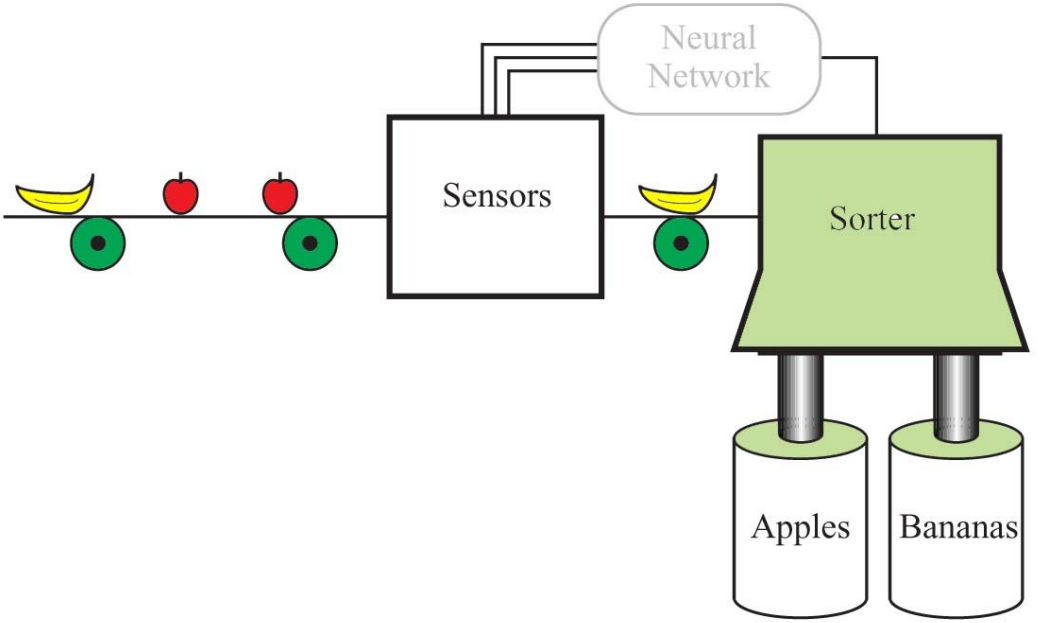
صورت مسئله

یک مثال گویا

حل یک مسئله‌ی ساده‌ی بازشناسی الگو با ۳ معماری مختلف شبکه عصبی



Apple/Banana Sorter

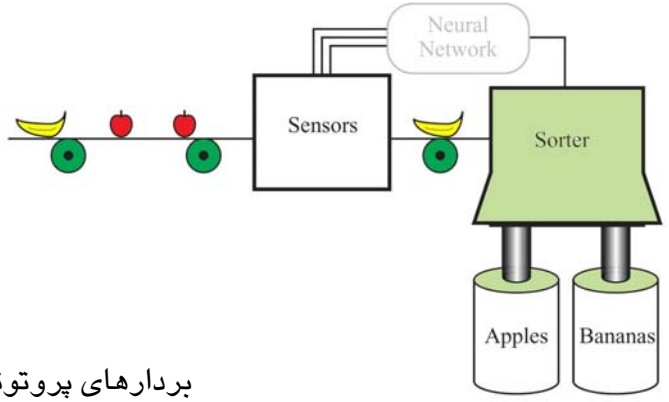


یک مثال گویا

دسته‌بندی موز / سیب

با استفاده از سه ویژگی میوه:

shape	**شکل:**	گرد (+1)	بیضوی (-1)
texture	**بافت:**	نرم (+1)	زبر (-1)
weight	**وزن:**	بالا (+1)	پایین (-1)



بردار اندازه‌گیری

$$\mathbf{p} = \begin{bmatrix} \text{shape} \\ \text{texture} \\ \text{weight} \end{bmatrix}$$

بردارهای پروتوتایپ (الگو)

$$\mathbf{p}_1 = \begin{bmatrix} -1 \\ +1 \\ -1 \end{bmatrix}$$

موز

$$a = +1$$

$$\mathbf{p}_2 = \begin{bmatrix} +1 \\ +1 \\ -1 \end{bmatrix}$$

سیب

$$a = -1$$

Prototype Vectors

Measurement Vector

$$\mathbf{p} = \begin{bmatrix} \text{shape} \\ \text{texture} \\ \text{weight} \end{bmatrix}$$

Shape: {1 : round ; -1 : eliptical}
Texture: {1 : smooth ; -1 : rough}
Weight: {1 : > 1 lb. ; -1 : < 1 lb.}

Prototype Banana

$$\mathbf{p}_1 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

Prototype Apple

$$\mathbf{p}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$$

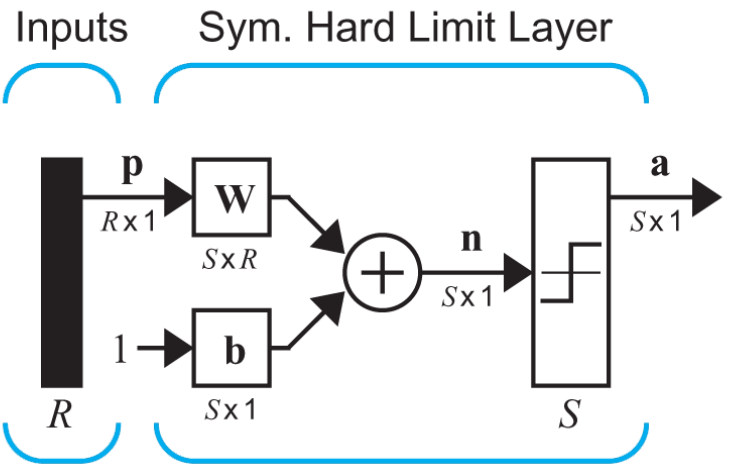
شبکه‌های عصبی مصنوعی

یک مثال گویا از شبکه‌ی عصبی

۲

پرسپترون

Perceptron

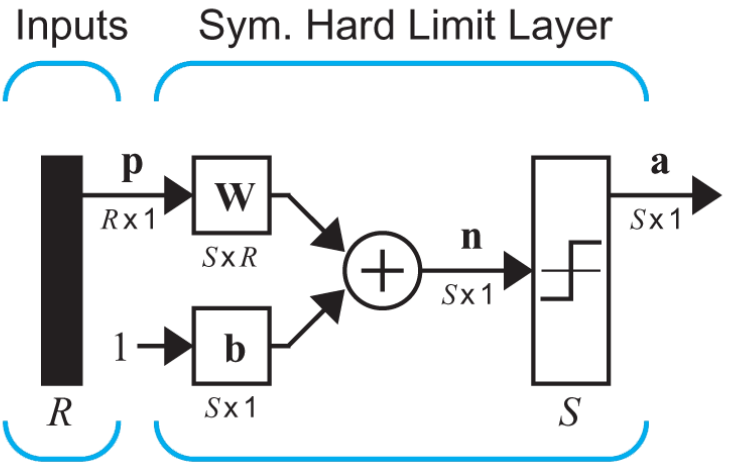


$$\mathbf{a} = \mathbf{hardlims}(\mathbf{Wp} + \mathbf{b})$$

پرسپترون

پرسپترون تک لایه

PERCEPTRON



$$\mathbf{p} = \begin{bmatrix} \text{shape} \\ \text{texture} \\ \text{weight} \end{bmatrix}$$

$R = 3$

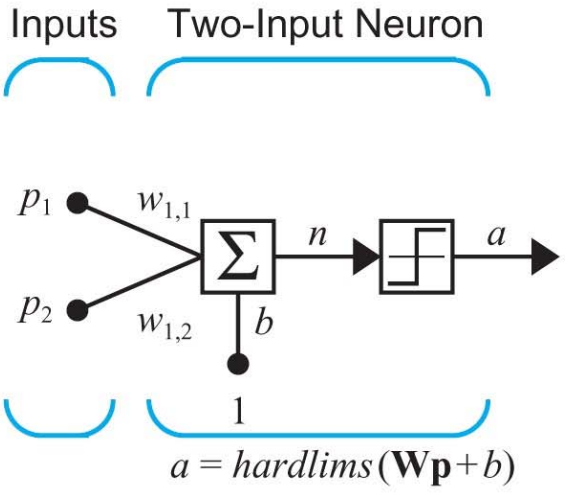
$S = 1$

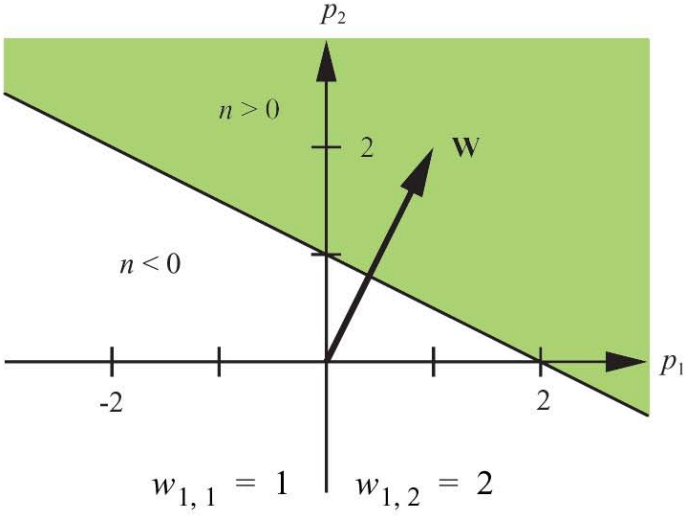
$$\mathbf{a} = \mathbf{hardlims}(\mathbf{Wp} + \mathbf{b})$$

خروجی: دو حالتی 1+ و 1-

تابع انتقال: حد سخت متقارن **hardlims**

Two-Input Case





$$a = hardlims(n) = hardlims(\begin{bmatrix}1 & 2\end{bmatrix}\mathbf{p} + (-2))$$

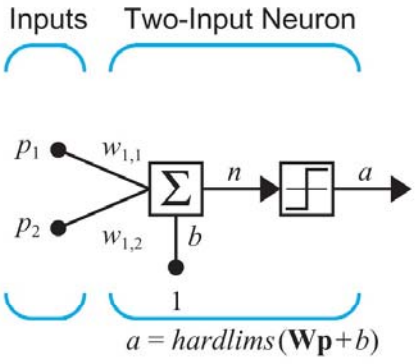
Decision Boundary

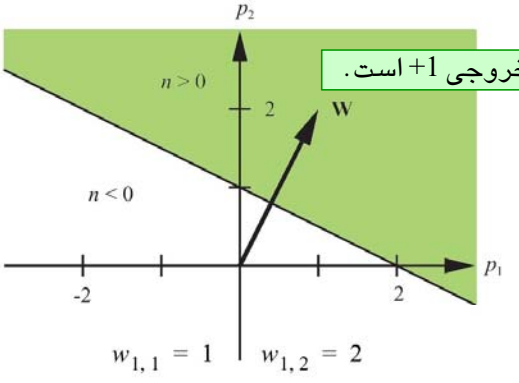
$$\mathbf{W}\mathbf{p} + b = 0 \qquad \begin{bmatrix}1 & 2\end{bmatrix}\mathbf{p} + (-2) = 0$$

پرسپترون

نرون دو ورودی

TWO-INPUT NEURON





جهت بردار وزن به سمت الگو با خروجی 1+ است.

$$a = hardlims(n) = hardlims(\begin{bmatrix}1 & 2\end{bmatrix}\mathbf{p} + (-2))$$

Decision Boundary

$$\mathbf{Wp} + b = 0 \qquad \begin{bmatrix}1 & 2\end{bmatrix}\mathbf{p} + (-2) = 0$$

نتیجه‌ی طبقه‌بندی توسط نرون دو ورودی بر روی صفحه قابل نمایش است.

مرز تصمیم:

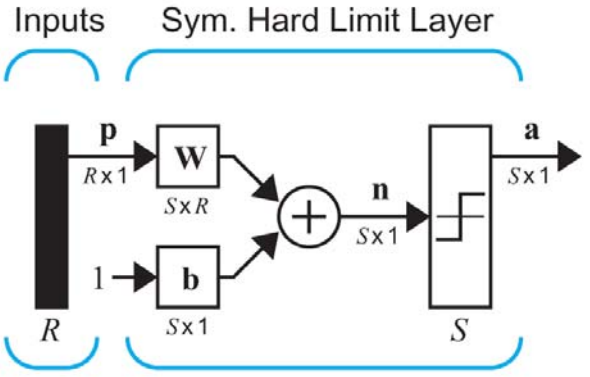
- همیشه بر بردار وزن **W** عمود است.
- مکان آن با تغییر بایاس شیفت پیدا می‌کند.

مرز تصمیم
Decision Boundary

$$n = \mathbf{Wp} + b = 0$$

پرسپترون

حالت کلی

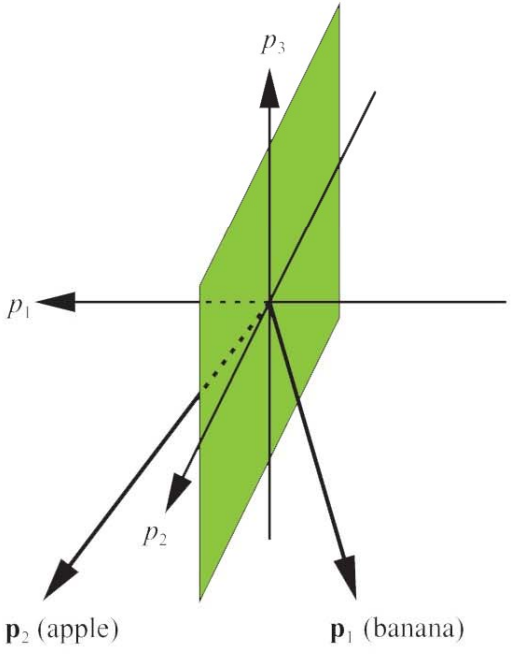


هر سطر ماتریس **W** یک بردار وزن است.
این بردار وزن، بردار نرمال یک خط / صفحه / ابرصفحه است.
⇓
برای هر سطر **W** یک مرز تصمیم داریم.

چون مرز باید خطی باشد ⇐ پرسپترون تک لایه فقط می‌تواند الگوهای جداپذیر خطی را طبقه‌بندی کند.

Apple/Banana Example

$$a = hardlims\left(\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + b\right)$$



The decision boundary should separate the prototype vectors.

$$p_1 = 0$$

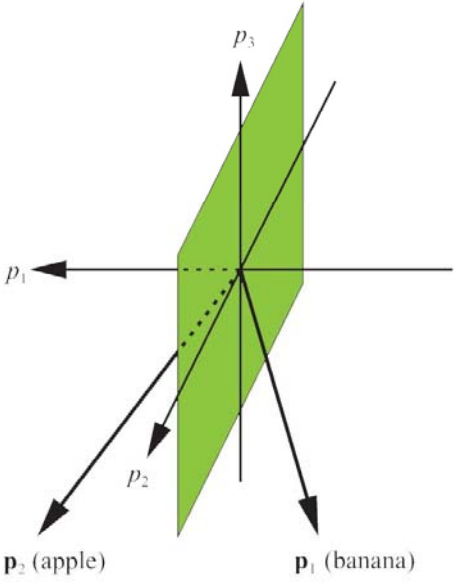
The weight vector should be orthogonal to the decision boundary, and should point in the direction of the vector which should produce an output of 1. The bias determines the position of the boundary

$$\begin{bmatrix} -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + 0 = 0$$

پرسپترون

برای مثال دسته‌بندی موز/سیب

$$a = hardlims\left(\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + b\right)$$



مرز تصمیم باید بردارهای پروتوتایپ را جدا کند:

$$p_1 = 0$$

بردار وزن باید عمود بر مرز تصمیم باشد
و به جهت برداری که خروجی $+1$ تولید می‌کند
اشاره کند.

بایاس موقعیت مرز را مشخص می‌کند.

$$\mathbf{W} = [-1 \quad 0 \quad 0], \quad b = 0$$

مرز تصمیم :

$$[-1 \quad 0 \quad 0]\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + 0 = 0$$

هر الگوی دیگری که وارد شود،
بسته به اینکه به کدام پروتوتایپ نزدیک‌تر باشد (در فضای اقلیدسی)،
در آن طبقه دسته‌بندی می‌شود.

Testing the Network

Banana:

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

Apple:

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

"Rough" Banana:

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

پرسپترون

برای مثال دسته‌بندی موز/سیب: تست شبکه

Banana:

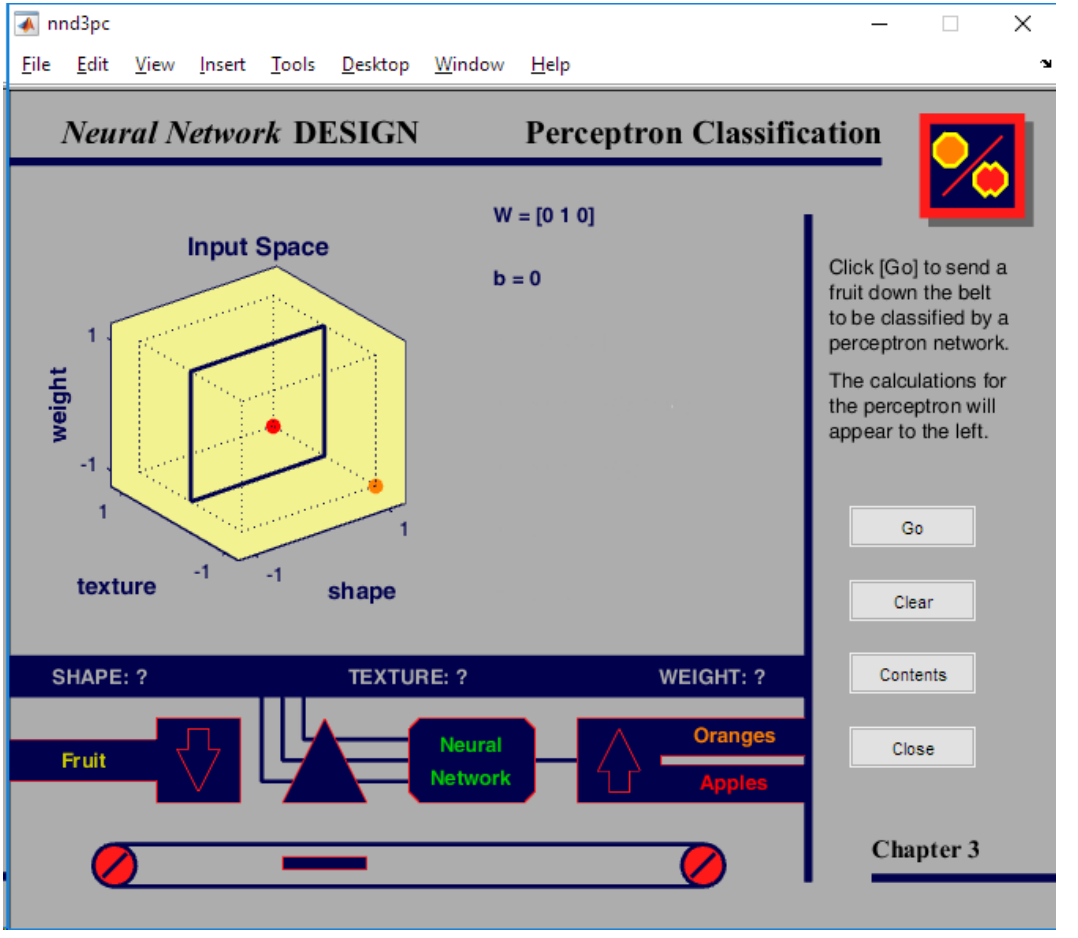
$$a = hardlims\left(\begin{bmatrix}-1 & 0 & 0\end{bmatrix}\begin{bmatrix}-1\\1\\-1\end{bmatrix} + 0\right) = 1(\text{banana})$$

Apple:

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

“Rough” Banana:

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


nnd3pc
File Edit View Insert Tools Desktop Window Help
Neural Network DESIGN
Perceptron Classification
W = [0 1 0]
b = 0
Input Space
weight
texture
shape
1
-1
Click [Go] to send a fruit down the belt to be classified by a perceptron network.
The calculations for the perceptron will appear to the left.
Go
Clear
Contents
Close
SHAPE: ?
TEXTURE: ?
WEIGHT: ?
Fruit
Neural Network
Oranges
Apples
Chapter 3

```
>> nnd3pc
```

پرسپترون

جمع‌بندی

✍ برای یافتن بردارهای وزن و بایاس، از روش‌های یادگیری استفاده می‌شود (فصل‌های ۴، ۷، ۱۰ و ۱۱)

✍ اگر دسته‌ها جداپذیر خطی نباشند، باید از شبکه‌ی پرسپترون چندلایه استفاده کنیم (فصل ۱۱)

پرسپترون

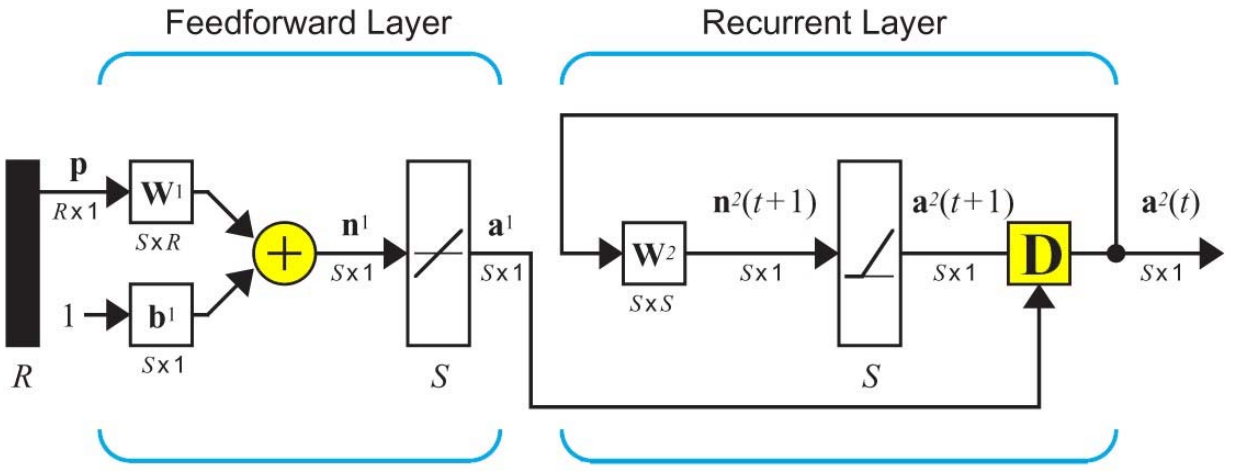
- شبکه‌ی پیش‌خور
- مرز تصمیم خطی
- یک نرون برای هر تصمیم

یک مثال گویا از شبکه‌ی عصبی

۳

شبکه‌ی همینگ

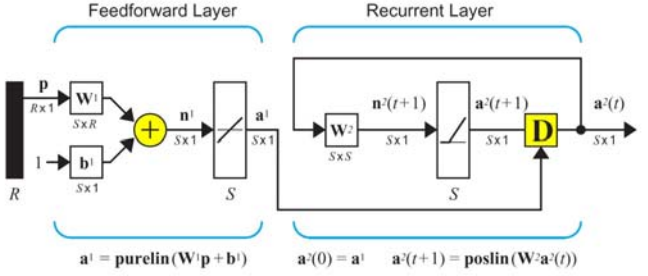
Hamming Network



شبکه‌ی همینگ

HAMMING NETWORK

شبکه‌ی همینگ: طراحی شده برای حل مسائل بازشناسی الگوی دودویی (عناصر ورودی دوحالته)



شامل دو لایه:

لایه‌ی پیش‌خور
Feedforward Layer

لایه‌ی بازگشتی
Recurrent Layer

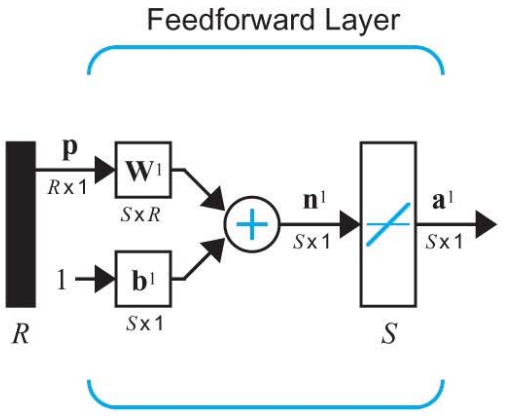
تعداد نرون لایه‌ی اول = تعداد نرون لایه‌ی دوم

هدف: بردار ورودی به کدام پروتوتایپ نزدیک‌تر است؟

خروجی: نمایش نتیجه در خروجی لایه‌ی بازگشتی

برای هر پروتوتایپ، یک نرون در لایه‌ی بازگشتی وجود دارد.
وقتی لایه‌ی بازگشتی همگرا شد، فقط یک نرون با خروجی غیرصفر وجود خواهد داشت.
(نرون مربوط به نزدیک‌ترین پروتوتایپ به بردار ورودی)

Feedforward Layer



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

For Banana/Apple Recognition

$$S = 2$$

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

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

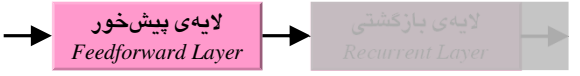
$$\mathbf{a}^1 = \mathbf{W}^1\mathbf{p} + \mathbf{b}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix}\mathbf{p} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^T\mathbf{p} + 3 \\ \mathbf{p}_2^T\mathbf{p} + 3 \end{bmatrix}$$

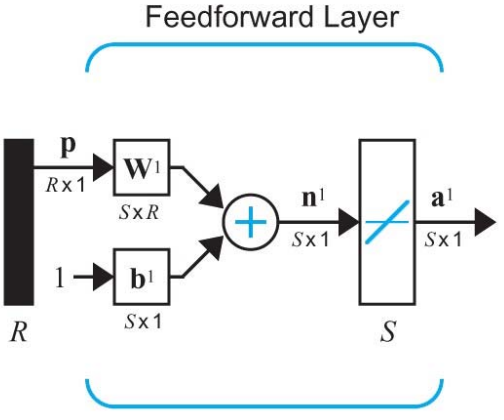
شبکه‌ی همینگ

لایه‌ی پیش‌خور

HAMMING NETWORK

لایه‌ی پیش‌خور: محاسبه‌ی همبستگی (ضرب داخلی) بین الگوهای پروتوتایپ و الگوی ورودی





برای مسئله‌ی بازشناسی موز / سیب:

$$S = 2$$

تابع انتقال خطی

ماتریس وزن:
هر سطر = ترانهاده‌ی یک پروتوتایپ

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

هر عنصر بردار بایاس = R
(برای جلوگیری از منفی شدن خروجی)

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

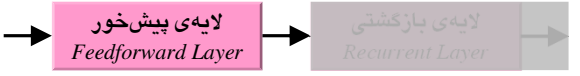
$$\mathbf{a}^1 = \mathbf{W}^1\mathbf{p} + \mathbf{b}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix}\mathbf{p} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^T\mathbf{p} + 3 \\ \mathbf{p}_2^T\mathbf{p} + 3 \end{bmatrix}$$

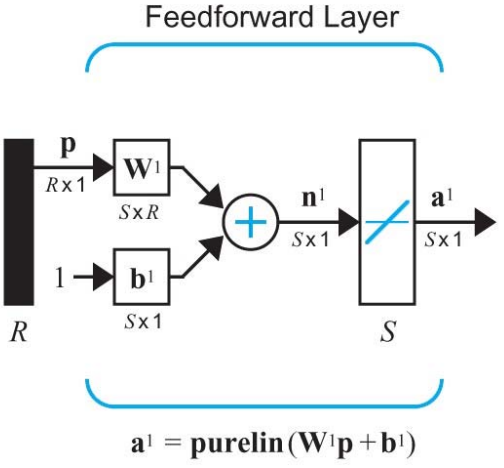
شبکه‌ی همینگ

لایه‌ی پیش‌خور

HAMMING NETWORK

لایه‌ی پیش‌خور: محاسبه‌ی همبستگی (ضرب داخلی) بین الگوهای پروتوتایپ و الگوی ورودی



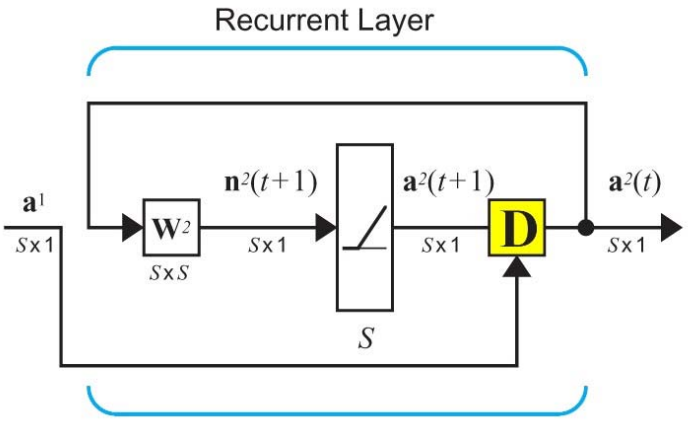


کمترین فاصله‌ی همینگ بین پروتوتایپ‌ها و بردار ورودی، مربوط به نرونی است که بزرگ‌ترین خروجی را دارد.

فاصله‌ی همینگ میان دو بردار دودویی = تعداد عناصر متفاوت آنها

خروجی لایه‌ی پیش‌خور = $2R$ منهای دو برابر فاصله‌های همینگ الگوهای پروتوتایپ از الگوهای ورودی

Recurrent Layer



$\mathbf{a}^2(0) = \mathbf{a}^1 \qquad \mathbf{a}^2(t+1) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(t))$

$$\mathbf{W}^2 = \begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix} \qquad \varepsilon < \frac{1}{S-1}$$

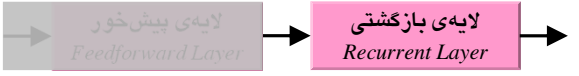
$$\mathbf{a}^2(t+1) = \mathbf{poslin}\left(\begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix}\mathbf{a}^2(t)\right) = \mathbf{poslin}\left(\begin{bmatrix} a_1^2(t) - \varepsilon a_2^2(t) \\ a_2^2(t) - \varepsilon a_1^2(t) \end{bmatrix}\right)$$

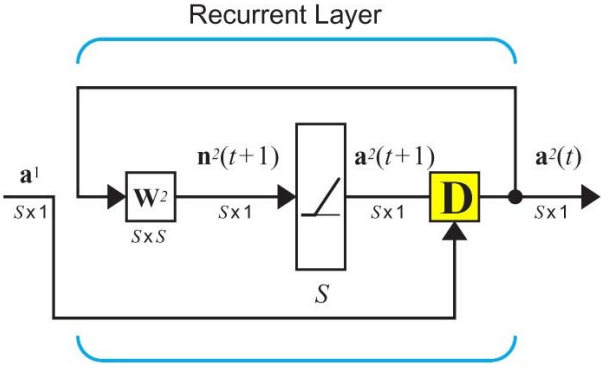
شبکه‌ی همینگ

لایه‌ی بازگشتی

HAMMING NETWORK

لایه‌ی بازگشتی: یک لایه‌ی رقابتی: رقابت نرون‌ها برای تعیین برنده (برنده همه را می‌خورد!)





$$\mathbf{a}^2(0) = \mathbf{a}^1 \qquad \mathbf{a}^2(t+1) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(t))$$

مقدار اولیه‌ی نرون‌ها =
خروجی لایه‌ی پیش‌خور (همبستگی ورودی با پروتوتایپ‌ها)

بعد از رقابت، تنها یک نرون با مقدار غیر صفر وجود دارد:
(نرون مشخص‌کننده‌ی طبقه‌ی بردار ورودی)

$$\mathbf{W}^2 = \begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix} \qquad \varepsilon < \frac{1}{S-1}$$

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

هر عنصر با کسر یکسانی از دیگران کم می‌شود ⇐
عنصر بزرگ‌تر کمتر کاهش می‌یابد و عنصر کوچک‌تر بیشتر ⇐
تفاوت میان عنصرهای بزرگ و کوچک افزایش می‌یابد ⇐
در نهایت، فقط یک نرون با بزرگ‌ترین مقدار غیر صفر باقی می‌ماند.

Hamming Operation

First Layer

Input (Rough Banana)

$$\mathbf{p} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

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

Hamming Operation

Second Layer

$$\mathbf{a}^2(1) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(0)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

$$\mathbf{a}^2(2) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(1)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ -1.5 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

شبکه‌ی همینگ

برای مسئله‌ی طبقه‌بندی موز / سیب

HAMMING NETWORK

لایه‌ی اول

لایه‌ی پیش‌خور
Feedforward Layer

لایه‌ی دوم

لایه‌ی بازگشتی
Recurrent Layer

$$\varepsilon = 0.5$$

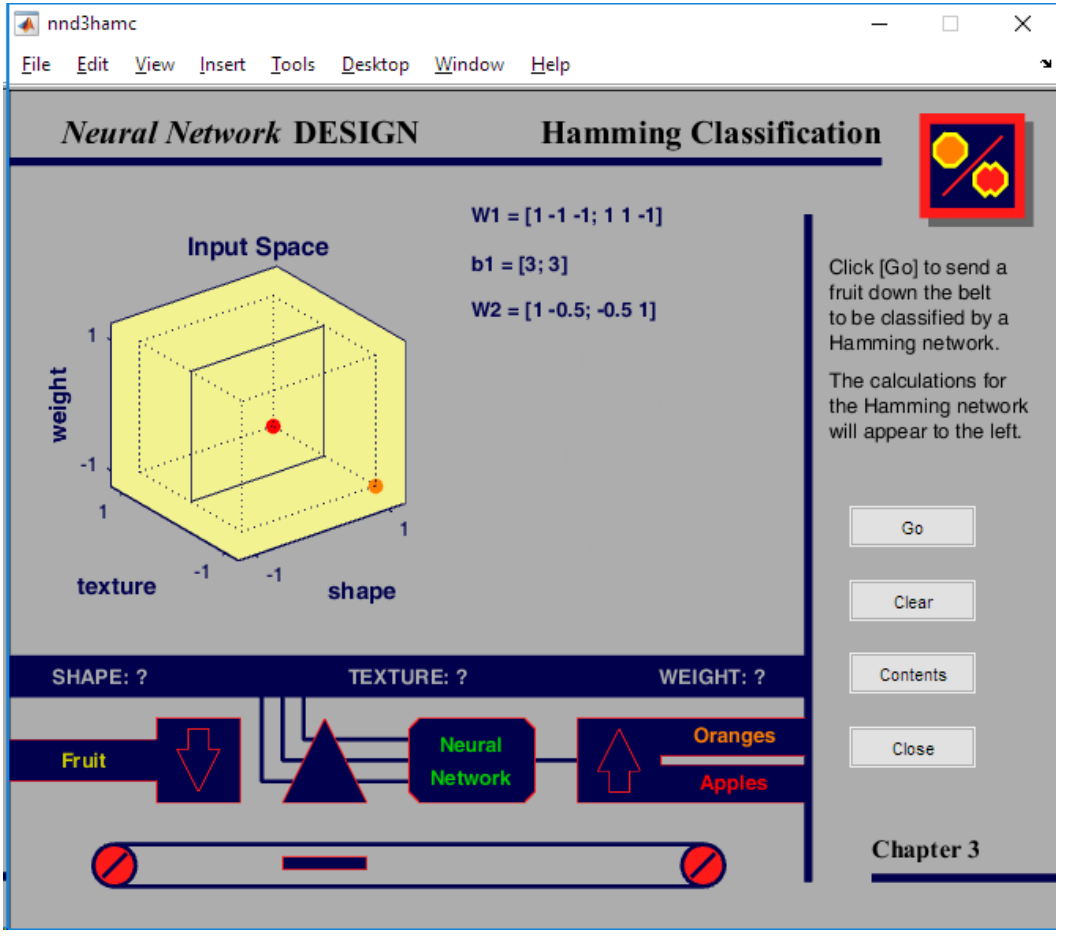
Input (Rough Banana)

$$\mathbf{p} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

$$a^1 = \begin{bmatrix} -1 & 1 & -1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1+3 \\ -1+3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

$$\mathbf{a}^2(1) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(0)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

$$\mathbf{a}^2(2) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(1)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ -1.5 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$


nnd3hamc
File Edit View Insert Tools Desktop Window Help
Neural Network DESIGN
Hamming Classification
W1 = [1 -1 -1; 1 1 -1]
b1 = [3; 3]
W2 = [1 -0.5; -0.5 1]
Input Space
weight
texture
shape
1
-1
Click [Go] to send a fruit down the belt to be classified by a Hamming network.
The calculations for the Hamming network will appear to the left.
Go
Clear
Contents
Close
SHAPE: ?
TEXTURE: ?
WEIGHT: ?
Fruit
Neural Network
Oranges
Apples
Chapter 3

```
>> nnd3hamc
```

شبکه‌ی همینگ

جمع‌بندی

HAMMING NETWORK

شبکه‌ی همینگ

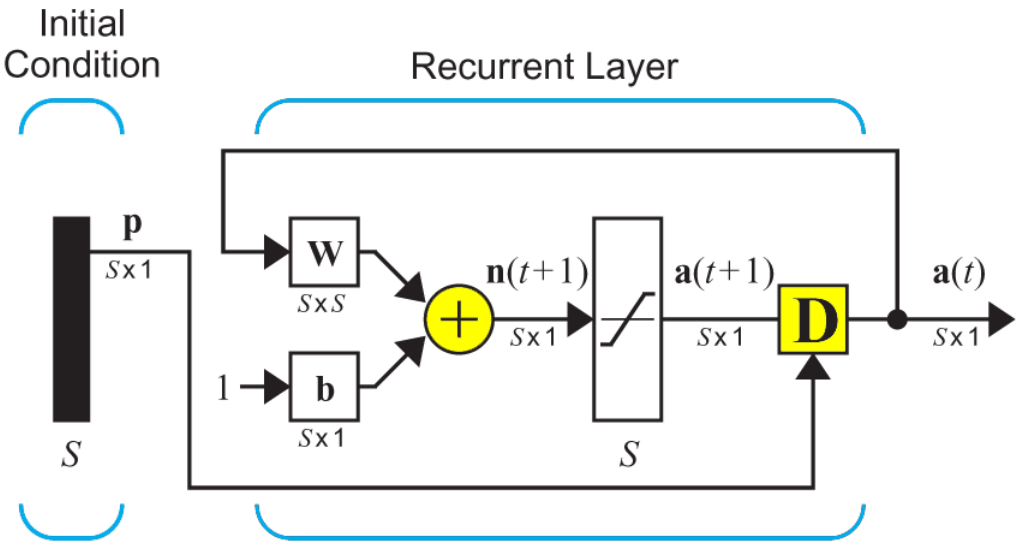
- شبکه‌ی رقابتی
- لایه‌ی اول: **تطابق الگو** (ضرب داخلی)
- لایه‌ی دوم: **رقابت** (برنده همه را می‌خورد!)
- تعداد نرون‌های هر لایه = تعداد الگوهای پروتوتایپ

یک مثال گویا از شبکه‌ی عصبی

۴

شبکه‌ی هاپفیلد

Hopfield Network

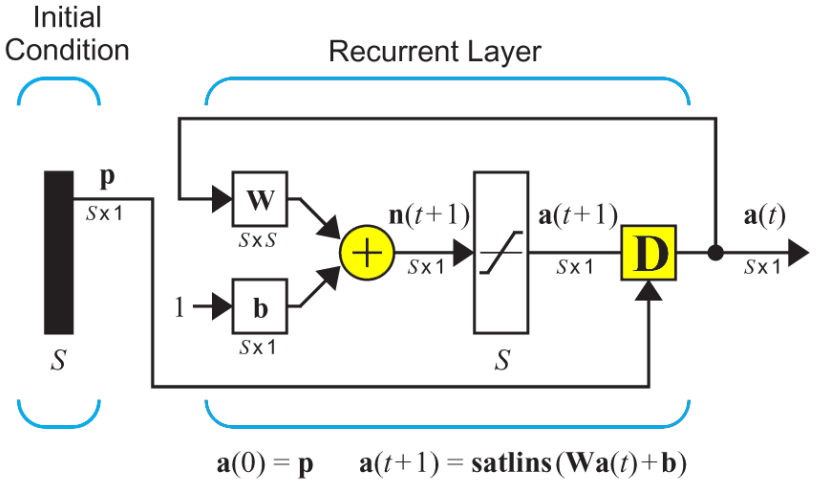


$$\mathbf{a}(0) = \mathbf{p} \qquad \mathbf{a}(t+1) = \mathbf{satlins}(\mathbf{W}\mathbf{a}(t)+\mathbf{b})$$

شبکه‌ی هاپفیلد

HOPFIELD NETWORK

شبکه‌ی هاپفیلد: یک شبکه‌ی بازگشتی که می‌تواند به طور کارآمد عملیات دو لایه‌ی شبکه‌ی همینگ را انجام دهد.



مقداردهی اولیه‌ی نرون‌ها با بردار ورودی

سپس شبکه وارد تکرار می‌شود تا زمانی که خروجی همگرا شود:
وقتی شبکه درست کار می‌کند: خروجی حاصل یکی از بردارهای پروتوتایپ است.

Apple/Banana Problem

$$\mathbf{W} = \begin{bmatrix} 1.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0.9 \\ -0.9 \end{bmatrix}$$

$$a_1(t+1) = satlins(1.2a_1(t))$$

$$a_2(t+1) = satlins(0.2a_2(t) + 0.9)$$

$$a_3(t+1) = \text{satlins}(0.2a_3(t) - 0.9)$$

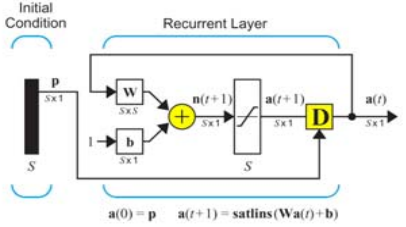
Test: “Rough” Banana

$$\mathbf{a}(0) = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{a}(1) = \begin{bmatrix} -1 \\ 0.7 \\ -1 \end{bmatrix} \quad \mathbf{a}(2) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{a}(3) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \text{ (Banana)}$$

شبکه‌ی هاپفیلد

مسئله‌ی موز / سیب

HOPFIELD NETWORK



$$\mathbf{W} = \begin{bmatrix} 1.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0.9 \\ -0.9 \end{bmatrix}$$

ضرب در یک ضریب بزرگ‌تر از یک:
بسته به مقدار اولیه به ± 1 همگرا می‌شود.

$$a_1(t+1) = satlins(1.2a_1(t))$$

افزایش می‌یابد تا به $+1$ همگرا شود (اشباع).

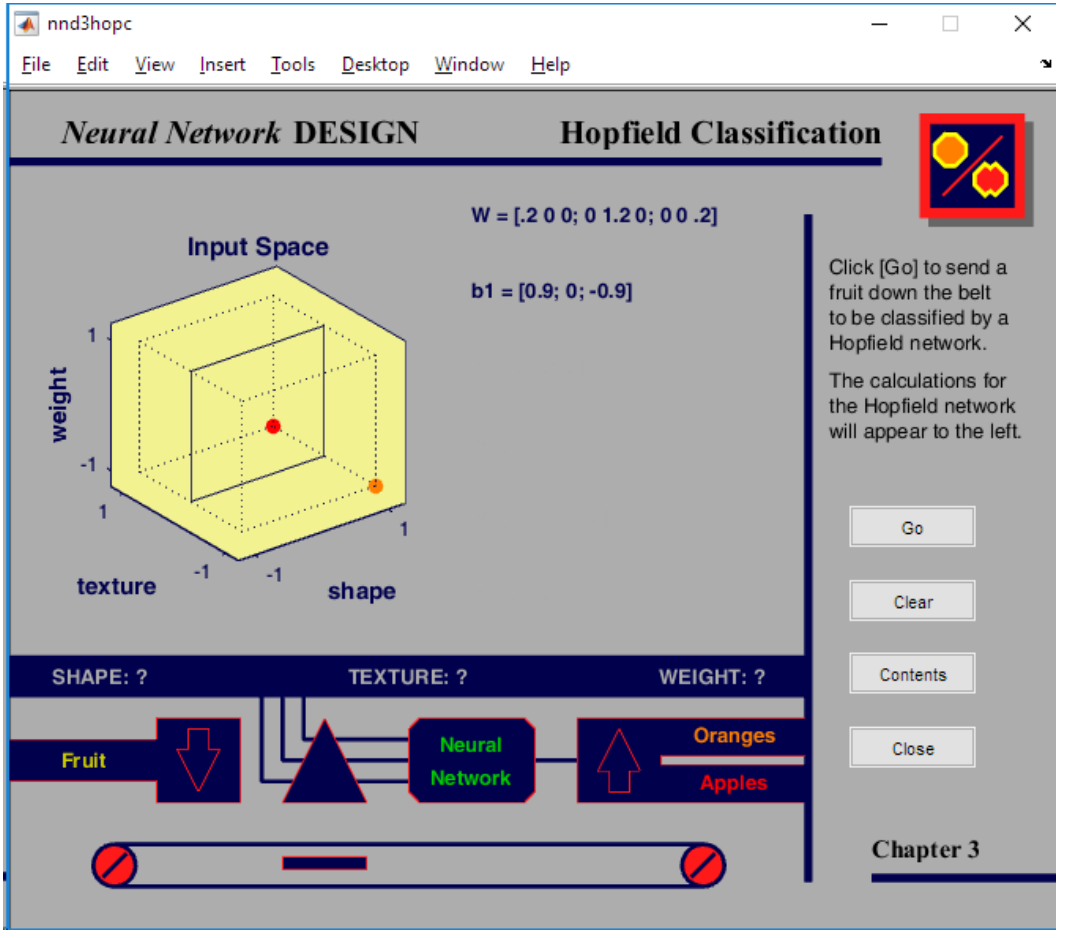
$$a_2(t+1) = satlins(0.2a_2(t) + 0.9)$$

کاهش می‌یابد تا به -1 همگرا شود (اشباع).

$$a_3(t+1) = \text{satlins}(0.2a_3(t) - 0.9)$$

Test: "Rough" Banana

$$\mathbf{a}(0) = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{a}(1) = \begin{bmatrix} -1 \\ 0.7 \\ -1 \end{bmatrix} \quad \mathbf{a}(2) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{a}(3) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \text{ (Banana)}$$


nnd3hopc
File Edit View Insert Tools Desktop Window Help
Neural Network DESIGN
Hopfield Classification
W = [.2 0 0; 0 1.2 0; 0 0 .2]
b1 = [0.9; 0; -0.9]
Input Space
weight
texture
shape
1
-1
Click [Go] to send a fruit down the belt to be classified by a Hopfield network.
The calculations for the Hopfield network will appear to the left.
Go
Clear
Contents
Close
SHAPE: ?
TEXTURE: ?
WEIGHT: ?
Fruit
Neural Network
Oranges
Apples
Chapter 3

>> nnd3hopc

شبکه‌ی هاپفیلد

HOPFIELD NETWORK

✍ برای یافتن ماتریس وزن از الگوریتم یادگیری فصل ۲۱ استفاده می‌شود.

شبکه‌ی هاپفیلد

- شبکه‌ی حافظه‌ی شرکت‌پذیر پویا
- خروجی شبکه به یک الگوی پروتوتایپ همگرا می‌شود.
- تعداد نرون‌ها = تعداد عناصر در هر الگوی پروتوتایپ

یک مثال گویا از شبکه‌ی عصبی

۵

خلاصه

Summary

- Perceptron
  - Feedforward Network
  - Linear Decision Boundary
  - One Neuron for Each Decision
- Hamming Network
  - Competitive Network
  - First Layer – Pattern Matching (Inner Product)
  - Second Layer – Competition (Winner-Take-All)
  - # Neurons = # Prototype Patterns
- Hopfield Network
  - Dynamic Associative Memory Network
  - Network Output Converges to a Prototype Pattern
  - # Neurons = # Elements in each Prototype Pattern

خلاصه

SUMMARY

پرسپترون

- شبکه‌ی پیش‌خور
- مرز تصمیم خطی
- یک نرون برای هر تصمیم

شبکه‌ی همینگ

- شبکه‌ی رقابتی
- لایه‌ی اول: **تطابق الگو** (ضرب داخلی)
- لایه‌ی دوم: **رقابت** (برنده همه را می‌خورد!)
- تعداد نرون‌های هر لایه = تعداد الگوهای پروتوتایپ

شبکه‌ی هاپفیلد

- شبکه‌ی حافظه‌ی شرکت‌پذیر پویا
- خروجی شبکه به یک الگوی پروتوتایپ همگرا می‌شود.
- تعداد نرون‌ها = تعداد عناصر در هر الگوی پروتوتایپ

تابع‌های انتقال

الف

Name	Input/Output Relation	Icon	MATLAB Function
Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$		satlin
Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = 1 \quad n > 1$		satlins

تابع‌های انتقال

ب

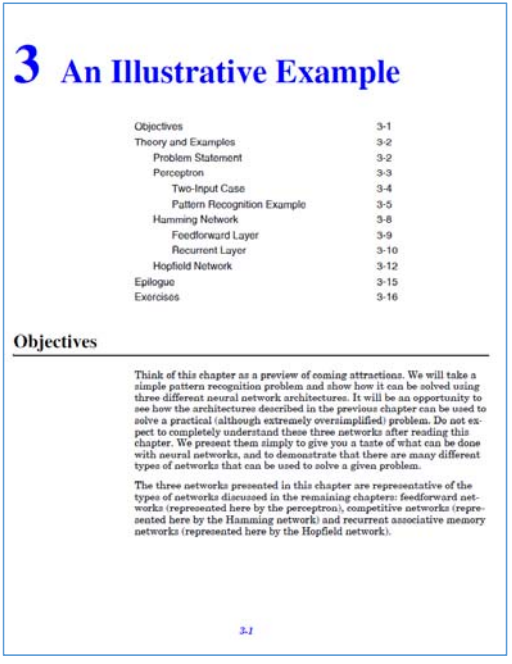
Name	Input/Output Relation	Icon	MATLAB Function
Log-Sigmoid	$a = \frac{1}{1+e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \le n$		poslin
Competitive	$a = 1$ neuron with max n $a = 0$ all other neurons		compet

یک مثال گویا از شبکه‌ی عصبی

۶

منابع

منبع اصلی



3 An Illustrative Example

Objectives

Think of this chapter as a preview of coming attractions. We will take a simple pattern recognition problem and show how it can be solved using three different neural network architectures. It will be an opportunity to see how the architectures described in the previous chapter can be used to solve a practical (although extremely oversimplified) problem. Do not expect to completely understand these three networks after reading this chapter. We present them simply to give you a taste of what can be done with neural networks, and to demonstrate that there are many different types of networks that can be used to solve a given problem.

The three networks presented in this chapter are representative of the types of networks discussed in the remaining chapters: feedforward networks (represented here by the perceptron), competitive networks (represented here by the Hamming network) and recurrent associative memory networks (represented here by the Hopfield network).

3-1

Martin T. Hagan, Howard B. Demuth, Mark H. Beale, Orlando De Jesus,
Neural Network Design,
2nd Edition, Martin Hagan, 2014.
Chapter 3

Online version can be downloaded from: http://hagan.okstate.edu/nnd.html