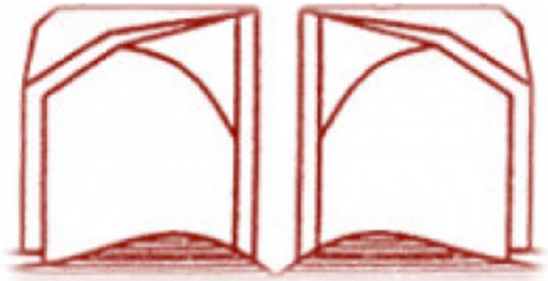


مبانی رایانش نرم

شبکه‌های عصبی: پرسپترون، آدالاین و شبکه‌های انجمنی

هادی ویسی

h.veisi@ut.ac.ir

دانشگاه تهران – دانشکده علوم و فنون نوین

فهرست

- شبکه پرسپترون
  - ساختار، الگوریتم، کاربردها و مثال
  - همگرایی قانون یادگیری
- شبکه آدالاین
  - ساختار، الگوریتم، کاربردها و مثال
  - قانون دلتا
- پیوند الگو و شبکه‌های انجمنی
  - الگوریتم‌های آموزش پیوند الگو: قانون هب و قانون دلتا
- شبکهٔ عصبی حافظهٔ دیگرانجمنی: ساختار، الگوریتم، کاربرد و مثال‌ها
- شبکهٔ عصبی حافظهٔ خودانجمنی: ساختار، الگوریتم، کاربرد و مثال‌ها
- شبکهٔ خودانجمن تکراری
- شبکهٔ هاپفیلد گسسته و پیوسته: ساختار، الگوریتم، کاربرد و مثال

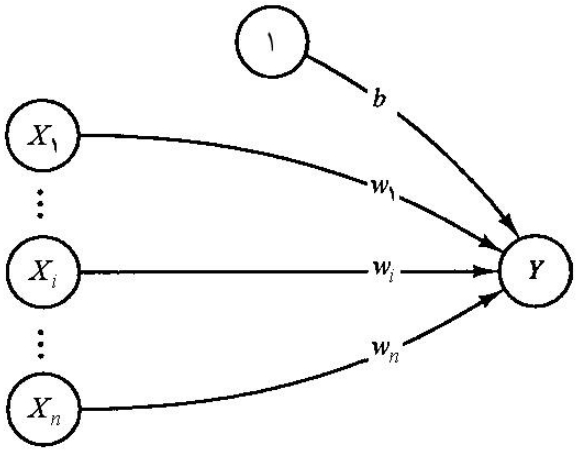
شبکه پرسپترون . . .

- جزو معروف‌ترین شبکه‌های عصبی است
  - حالت چند لایه آنها از پرکاربردترین شبکه‌های عصبی هستند
  - بیشترین اثرگذاری بر شبکه‌های عصبی اولیه
  - روزنبلات در سال ۱۹۶۲ و مینسکی و پاپرت در سال‌های ۱۹۶۹ و ۱۹۸۸
- قانون یادگیری قوی‌تر نسبت به قانون هب
  - یادگیری همراه با تکرار
    - در قانون هب، فقط یک بار (بدون تکرار) داده‌های آموزش به شبکه داده می‌شد
  - یادگیری پرسپترون شبیه قانون هب، تفاوت عمده: وزن‌ها فقط زمانی تغییر می‌کند که پاسخ شبکه به ازای آن ورودی دارای خطا باشد
    - خطا = خروجی محاسبه شده توسط شبکه با مقدار هدف یکی نباشد

شبکه پرسپترون: ساختار . . .

- ساختاراولیه
  - سه لایه نرون (واحدهای حسی، واحدهای پیونددهنده، و واحد پاسخ)
    - فقط وزن‌های بین لایه‌های دوم و سوم آموزش داده می‌شود
    - خروجی واحدهای پیونددهنده به واحدهای پاسخ یک بردار دودویی است
    - عملاً یک شبکه یک لایه است
  - مدل تقریبی شبکیه چشم

- ساختار برای دسته‌بندی الگو
  - متعلق بودن به دسته با پاسخ ۱+
  - متعلق نبودن با پاسخ ۱-



شبکه پرسپترون: الگوریتم . . .

- مرحلهٔ ۰ – مقداردهی اولیه به وزن‌ها و بایاس (مقدار صفر)
  تعیین نرخ یادگیری $0<\alpha\leq 1$ (مقدار ۱)
- مرحلهٔ ۱ – تا زمانی‌که شرایط توقف برقرار نیست، مراحل ۲ تا ۶ را انجام دهید
- مرحلهٔ ۲ – انجام مراحل ۳ تا ۵ برای هر جفت داده آموزش $s:t$
- مرحلهٔ ۳ – فعال‌سازی‌های واحدهای ورودی را مشخص کنید: $x_i = s_i$
- مرحلهٔ ۴ – پاسخ واحد خروجی را محاسبه کنید:

الگوریتم تکراری

$$y\_in = b + \sum_i x_i w_i$$

$$y = \begin{cases} 1 & if \quad y\_in > \theta \\ 0 & if \quad -\theta \leq y\_in \leq \theta \\ -1 & if \quad y\_in < -\theta \end{cases}$$

جانشینی بایاس و آستانه؟

۱ = تعلق به دسته ۱- = عدم تعلق به دسته ۰ = ناحیه عدم تصمیم‌گیری (2θ)

شبکه پرسپترون: الگوریتم . . .

- مرحلهٔ ۵- اگر خطایی رخ داده است، وزن‌ها و بایاس را به‌روز کنید.

به‌روز کردن مشروط وزن‌ها

اگر $y \neq t$ است، آنگاه:

$$w_i(new) = w_i(old) + \alpha x_i t$$
$$b(new) = b(old) + \alpha t$$

در غیراین‌صورت:

$$w_i(new) = w_i(old)$$
$$b(new) = b(old)$$

- مرحلهٔ ۶- شرایط توقف را آزمایش کنید:
  - اگر در مرحلهٔ ۲ هیچ وزنی تغییر نکرد، الگوریتم را متوقف کنید، در غیراین‌صورت ادامه دهید.

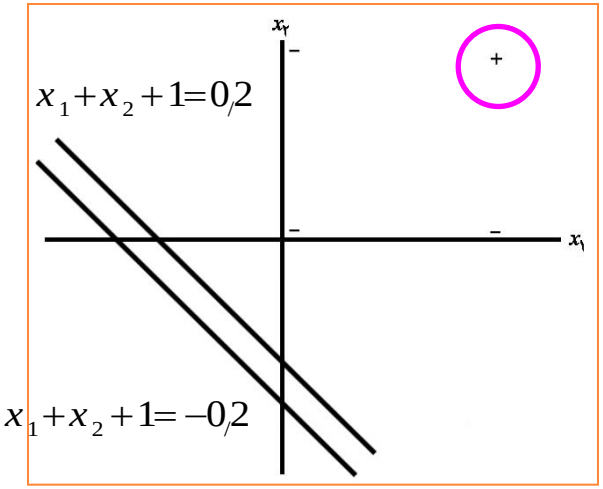
خطا = برابر نبودن پاسخ شبکه و مقدار هدف

نرخ یادگیری

شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - وزن‌های اولیه و بایاس را صفر؛ نرخ اولیه یادگیری = ۱؛ آستانه =۰.۲
  - ارائه ورودی اول

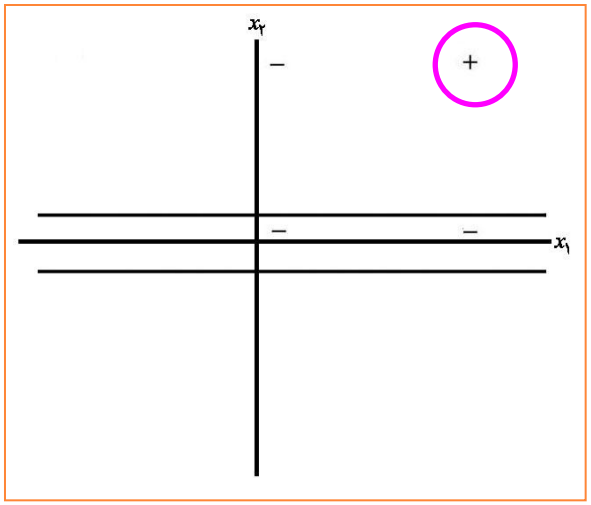
INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									0	0	0
1	1	1	0	0	1	1	1	1	1	1	1



شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - ارائه دومین ورودی

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									1	1	1
1	0	1	2	1	−1	−1	0	−1	0	1	0



$x_2 = 0.2$

$x_2 = -0.2$

شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های **دودویی** و هدف‌های **دوقطبی** . . .
  - ارائه سومین ورودی

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									0	1	0
0	1	1	1	1	−1	0	−1	−1	0	0	−1

  - ارائه چهارمین ورودی
    - با توجه به برابر بودن پاسخ شبکه و مقدار هدف، وزن‌ها تغییری نمی‌کنند

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									0	0	−1
0	0	1	−1	−1	−1	0	0	0	0	0	−1

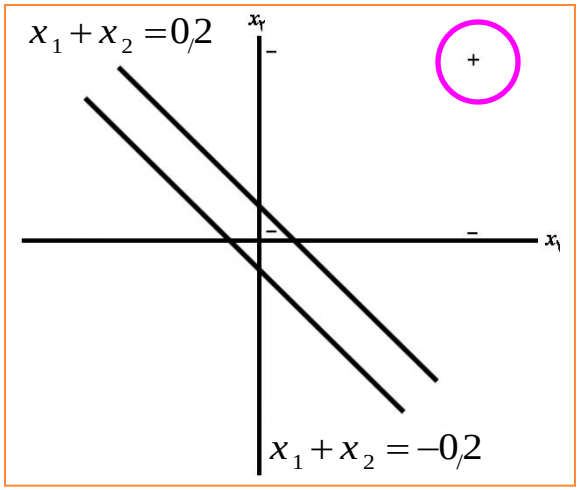
کامل شدن اولین دور آموزش (Epoch)

شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - نیاز به تکرار؟؟ صحیح نبودن پاسخ برای اولین الگوی ورودی
  - تکراری بودن فرآیند آموزش (Iterative)

- دومین دور آموزش –ارائه اولین ورودی

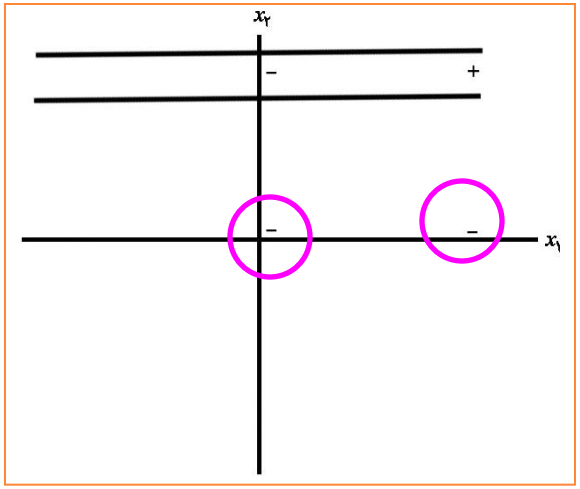
INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
$(x_1$	x_2	$1)$	y_in	y	t	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	w_2	$b)$
									$(0$	0	$-1)$
$(1$	1	$1)$	-1	-1	1	$(1$	1	$1)$	$(1$	1	$0)$



شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - دومین دور آموزش –ارائه دومین ورودی

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
$(x_1$	x_2	$1)$	y_in	y	t	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	w_2	$b)$
									$(1$	1	$0)$
$(1$	0	$1)$	1	1	-1	$(-1$	0	$-1)$	$(0$	1	$-1)$



$$x_2 - 1 = 0/2$$

$$x_2 - 1 = -0/2$$

شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - دومین دور آموزش –ارائه سومین ورودی
    - پاسخ برای تمام ورودی‌ها منفی

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									0	1	–1
0	1	1	0	0	–1	0	–1	–1	0	0	–2

  - دومین دور آموزش –ارائه چهارمین ورودی
    - پاسخ برای تمام ورودی‌ها منفی

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									0	0	–2
0	0	1	–2	–1	–1	0	0	0	0	0	–2

کامل شدن دومین دور آموزش (Epoch)

شبکه پرسپترون: کاربرد . . .

- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - سومین دور آموزش

INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
x_1	x_2	1	y_in	y	t	Δw_1	Δw_2	Δb	w_1	w_2	b
									0	0	−2
1	1	1	−2	−1	1	1	1	1	1	1	−1
1	0	1	0	0	−1	−1	0	−1	0	1	−2
0	1	1	−1	−1	−1	0	0	0	0	1	−2
0	0	1	−2	−1	−1	0	0	0	0	1	−2

  - چهارمین دور آموزش

1	1	1	−1	−1	1	1	1	1	1	2	−1
1	0	1	0	0	−1	−1	0	−1	0	2	−2
0	1	1	0	0	−1	0	−1	−1	0	1	−3
0	0	1	−3	−1	−1	0	0	0	0	1	−3

شبکه پرسپترون: کاربرد . . .

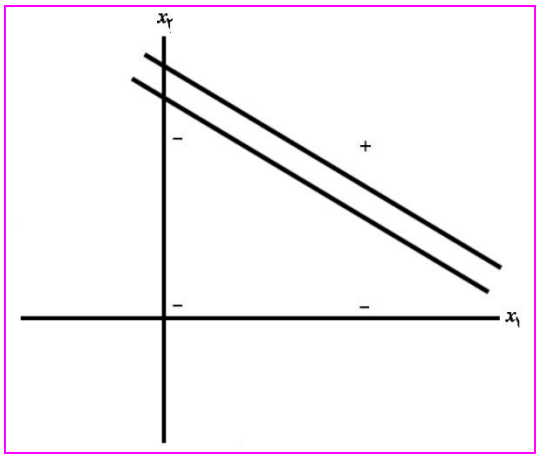
- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - پنجمین، ششمین، .... دور آموزش

$$\begin{array}{cccccccccc} (1 & 1 & 1) & 0 & 0 & 1 & (1 & 1 & 1) & (3 & 3 & -3) \\ (1 & 0 & 1) & 0 & 0 & -1 & (-1 & 0 & -1) & (2 & 3 & -4) \\ (0 & 1 & 1) & -1 & -1 & -1 & (0 & 0 & 0) & (2 & 3 & -4) \\ (0 & 0 & 1) & -4 & -1 & -1 & (0 & 0 & 0) & (2 & 3 & -4) \end{array}$$

  - نهمین دور آموزش

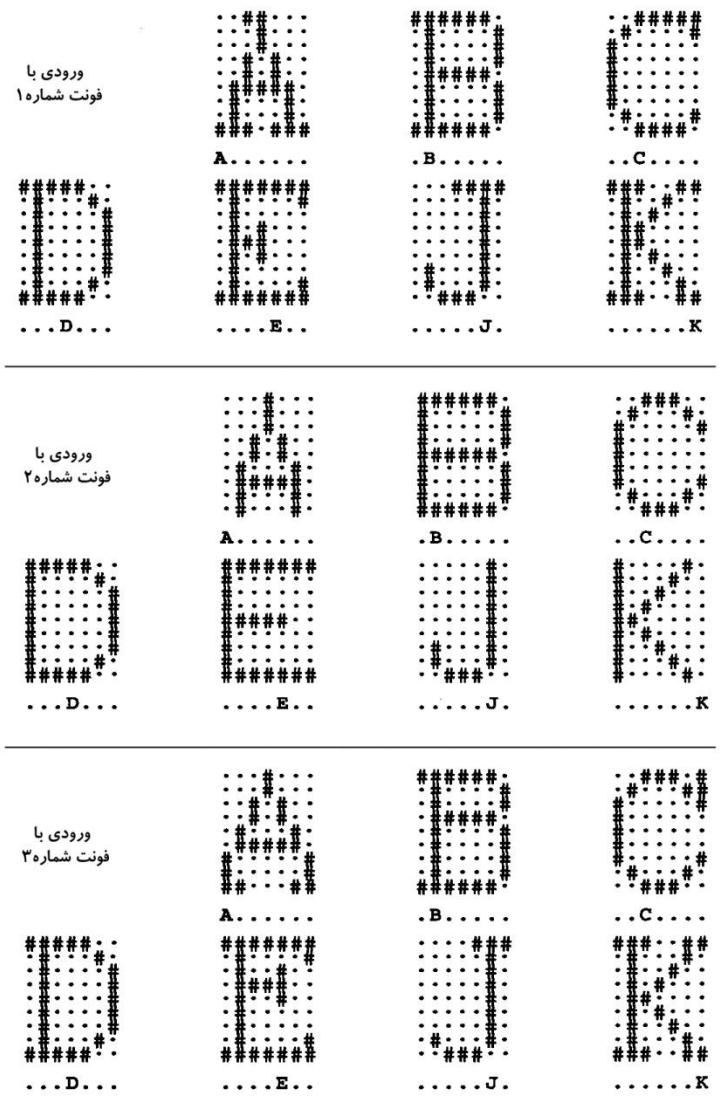
$$\begin{array}{cccccccccc} (1 & 1 & 1) & 1 & 1 & 1 & (0 & 0 & 0) & (2 & 3 & -4) \\ (1 & 0 & 1) & -2 & -1 & -1 & (0 & 0 & 0) & (2 & 3 & -4) \\ (0 & 1 & 1) & -1 & -1 & -1 & (0 & 0 & 0) & (2 & 3 & -4) \\ (0 & 0 & 1) & -4 & -1 & -1 & (0 & 0 & 0) & (2 & 3 & -4) \end{array}$$

  - دهمین دور آموزش
    - عدم تغییر وزن‌ها = توقف الگوریتم
    - همگرایی وزن‌ها



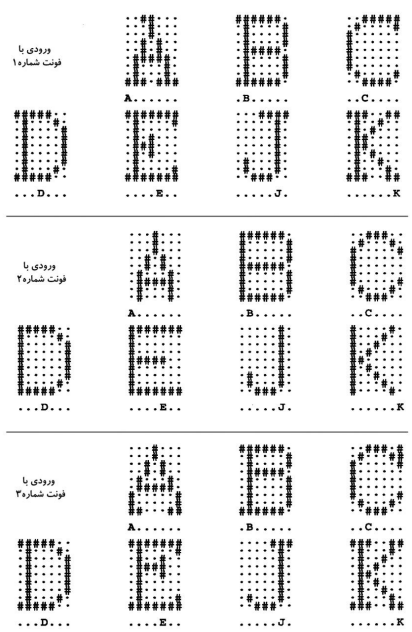
$$\begin{cases} 2x_1 + 3x_2 - 4 > 0{,}2 \Rightarrow x_2 = -\dfrac{2}{3}x_1 + \dfrac{7}{5} \\ 2x_1 + 3x_2 - 4 < -0{,}2 \Rightarrow x_2 = -\dfrac{2}{3}x_1 + \dfrac{19}{15} \end{cases}$$

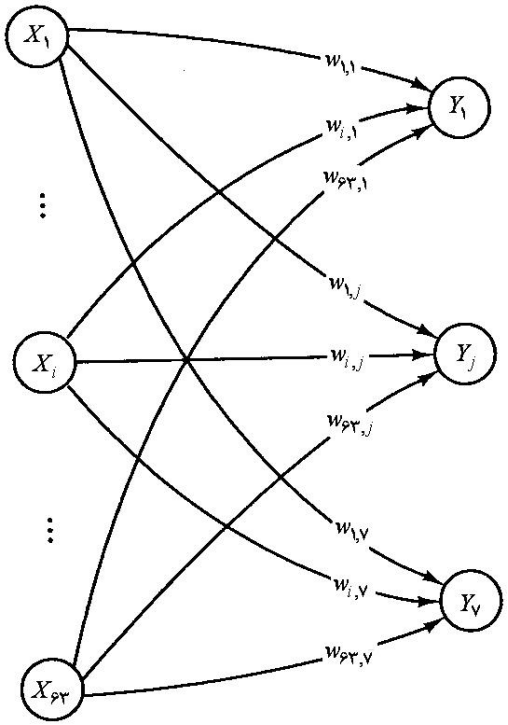
شبکه پرسپترون: مثال . . .



- بازشناسی نویسه . . .
  - تشخیص حرف A از حرف غیر A
  - ۳ نوع فونت
    - ۳ نوع A= خروجی شبکه معادل با تعلق به دسته
    - ۱۸ نویسه غیر A= خروجی معادل با عدم تعلق به دسته
  - تعمیم برای سایر نویسه‌ها؟؟
    - یک شبکه جداگانه برای هر نویسه
    - شبکه‌ای با چندین (به تعداد نویسه‌ها) خروجی

شبکه پرسپترون: مثال . . .





- بازشناسی نویسه . . .
  - نمایش دوقطبی
  - ۶۳ واحد ورودی (هر کدام یک پیکسل)
  - ۷ واحد خروجی (هر کدام یک نویسه)
    - بردار خروجی برای نمونه‌های A

$$(A.....) \Rightarrow (1,-1,-1,-1,-1,-1,-1)$$

    - بردار خروجی برای نمونه‌های B

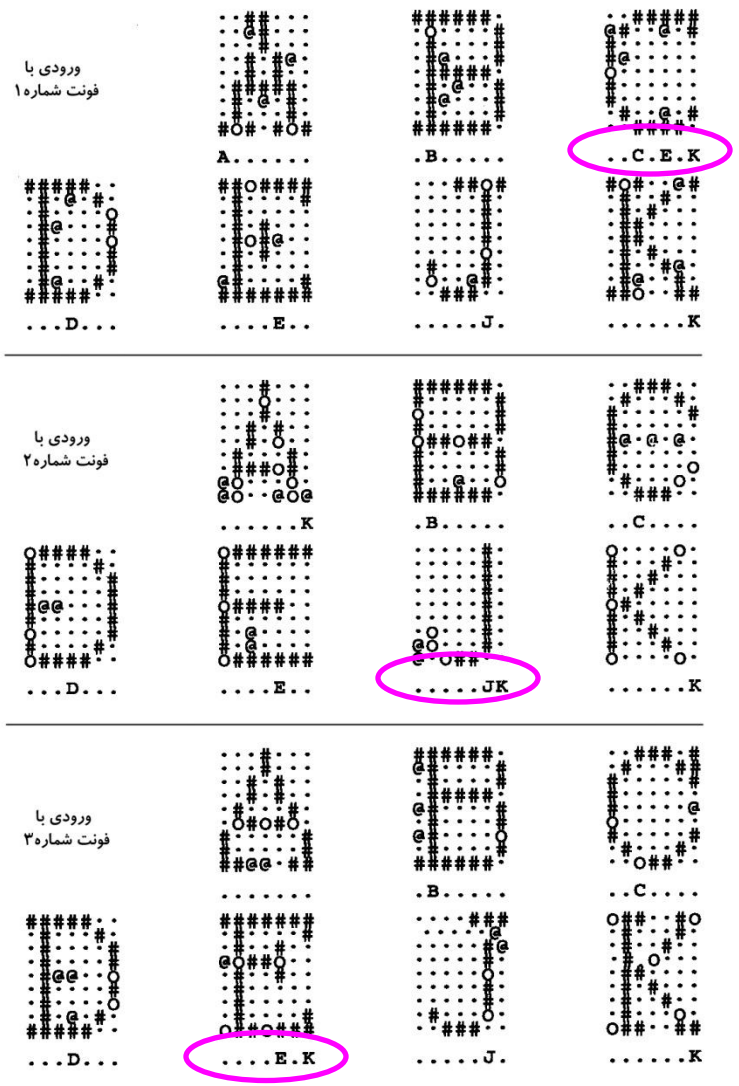
$$(.B....) \Rightarrow (-1,1,-1,-1,-1,-1,-1)$$

- دسته‌بندی درست داده‌های آموزش داده شده

شبکه پرسپترون: مثال

- بازشناسی نویسه . . .
  - ارزیابی شبکه با ورودی‌های نویزی
    - بردارهای ورودی شبیه به بردارهای آموزش
      - و نه دقیقاً مانند آنها

- جایگزینی پیکسل «غیرفعال» با «فعال» = @
- جایگزینی پیکسل «فعال» با «غیرفعال» = O



شبکه پرسپترون: همگرایی قانون یادگیری

- **قضیه**
  - اگر بردار وزن w^* وجود داشته باشد به‌طوری‌که برای تمام p‌ها داشته باشیم:

$$f(x(p).\ w^*) = t(p)$$

آنگاه برای هر بردار اولیه w، قانون یادگیری پرسپترون به بردار وزنی نزدیک می‌شود (نه الزاماً منحصر به فرد و نه الزاماً w^*) که برای تمام الگوهای آموزش پاسخ صحیحی می‌دهد و این کار در مراحلی با تعداد متناهی انجام می‌شود.

- p= تعداد بردارهای ورودی آموزش
- $x(p)$ = بردارهای ورودی آموزش
- $t(p)$= مقدار هدف معادل بردارهای ورودی آموزش (دوقطبی)
- f = تابع فعال‌سازی خروجی

شبکه پرسپترون: نکات تکمیلی

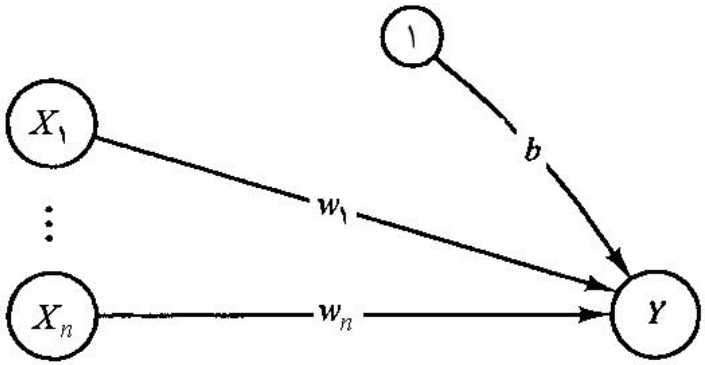
- مقداردهی نرخ یادگیری
  - مقدار ثابت غیرمنفی
  - مقدار $1/\|x\|$ ؛ تا تغییر وزن یک بردار واحد باشد
  - مقدار $(x.w)/\|x\|^2$
- مقادیر اولیه وزن‌ها
  - مقدار ثابت صفر
  - یک الگوی آموزش اختیاری
  - مقادیر تصادفی کوچکی

شبکه آدالاین ...

- آدالاین = نرون خط وفقی(ADAptive LInear Neuron)
  - توسط ویدور و هاف در سال ۱۹۶۰
  - دارای قانون یادگیری متفاوت با هب و پرسپترون
  - قانون یادگیری= قانون دلتا = قانون میانگین مربعات کمینه (LMS) = قانون ویدرو-هاف
    - میانگین مربعات خطای بین مقدار خروجی شبکه و مقدار هدف در هر مرحله از آموزش کاهش یابد
  - استفاده از فعال‌سازی‌های دوقطبی برای سیگنال‌های وروی و خروجی
  - تابع فعال‌سازی خروجی = تابع همانی

شبکه آدالاین: ساختار

- ساختار مشابه با سایر شبکه‌های قبلی
  - چند ورودی
  - بایاس = ورودی برابر با ۱



- قابلیت توسعه به حالت چندلایه = شبکهٔ مادالاین

شبکه آدالاین: الگوریتم . . .

- مرحلهٔ ۰- مقداردهی اولیه به وزن‌ها (مقادیر تصادفی کوچک)
  مقداردهی به نرخ یادگیری
- مرحلهٔ ۱- تا زمانی‌که شرایط توقف برقرار نیست، مراحل ۲ تا ۶ را انجام دهید.
- مرحلهٔ ۲- برای هر جفت آموزش دوقطبی $s:t$ مراحل ۳ تا ۵ را انجام دهید.
- مرحلهٔ ۳- فعال‌سازی‌های واحدهای ورودی را مشخص کنید: $x_i = s_i \qquad i = 1,\ldots,n$
- مرحلهٔ ۴- مقدار ورودی شبکه را به واحد خروجی محاسبه کنید: $y\_in = b + \sum_i x_i w_i$
- مرحلهٔ ۵- مقادیر وزن‌ها و بایاس را به‌روز کنید:
$$\begin{cases} b(new) = b(old) + \alpha.(t - y\_in) \\ w_i(new) = w_i(old) + \alpha.(t - y\_in).x_i \end{cases}$$
- مرحلهٔ ۶- شرایط توقف را آزمایش کنید:
  اگر بزرگ‌ترین تغییر وزنی که در مرحلهٔ ۲ رخ داده است از یک مقدار کوچک کم‌تر باشد، الگوریتم را متوقف کنید، وگرنه ادامه دهید.

شبکه آدالاین: الگوریتم

- **تفاوت یادگیری آدالاین با یادگیری پرسپترون و هب**
  - تغییر وزن‌ها متناسب با میزان تفاوت پاسخ شبکه به یک ورودی و مقدار هدف متناظر این ورودی است.
  - دربرگیرنده مفهوم خطا (که در یادگیری پرسپترون نیز وجود دارد)

- **نرخ یادگیری**
  - تاثیر بر سرعت و روند همگرایی الگوریتم
  - نیاز به اختصاص مقدار مناسب
  - دارای کران بالا از نظر تئوری
  - روش: ابتدا مقدار را کوچک فرض کرده (مثلاً ۰.۱) و به مرور مقدار آن را بزرگ کنیم
  - اگر مقدار خیلی بزرگی باشد، فرآیند یادگیری همگرا نخواهد بود
  - اگر مقدار بسیار کوچکی باشد، یادگیری بسیار کند می‌شود

شبکه آدالاین: کاربرد . . .

- تابع AND: ورودی‌های دودویی، هدف‌های دوقطبی
  - شبکه بعد از آموزش

x_1	x_2	t
1	1	1
1	0	−1
0	1	−1
0	0	−1

$$w_1 = 1 \quad w_2 = 1 \quad w_0 = -\frac{3}{2}$$

$$x_1 + x_2 - \frac{3}{2} = 0$$

  - مربعات خطا برای چهار الگوی آموزش با این وزن‌ها = ۱

$$e = E\{(\hat{t} - t)^2\} = \sum_{p=1}^{4} [\{x_1(p).w_1 + x_2(p).w_2 + w_0\} - t(p)]^2$$

شبکه آدالاین: قانون یادگیری

قانون دلتا

- کمینه کردن خطای بین خروجی شبکه و مقدار هدف متناظر
- خطا = مربعات تفاضل

$$E = (t - y\_in)^2$$

$$y\_in = \sum_{i=1}^{n} x_i w_i$$

- گرادیان تابع خطا= مشتق‌های جزئی خطا نسبت به هر یک از وزن‌ها
- گرادیان بیانگر جهت سریع‌ترین رشد خطا
- جهت مخالف گرادیان = سریع‌ترین کاهش خطا

$$-\frac{\partial E}{\partial w_I} = -2(t - y\_in)\frac{\partial y\_in}{\partial w_I} = -2(t - y\_in)x_I$$

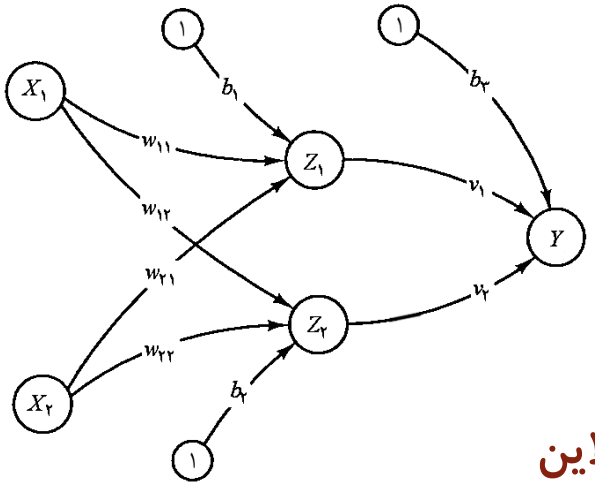
$$\Delta w_I = \alpha(t - y\_in)x_I$$

شبکه مادالاین

حالت چند لایه آدالاین

- ترکیب چندین واحد آدالاین در یک شبکهٔ یک لایه با هم = عدم تغییر در فرآیند آموزش
- برای بیش از یک لایه = نیاز به آموزش متفاوت
- شبکه چند لایه = افزایش قابلیت‌های محاسباتی شبکه = حل مسائل پیچیده‌تر

ساختار



الگوریتم

- MRI = شیوهٔ اصلی آموزش مادالاین
  - آموزش وزن‌های لایه اول و ثابت گرفتن وزن‌های لایه دوم (محاسبه به صورت شهودی)
- MRII: روشی دیگر

H. Veisi (h.veisi@ut.ac.ir)

پیوند الگو

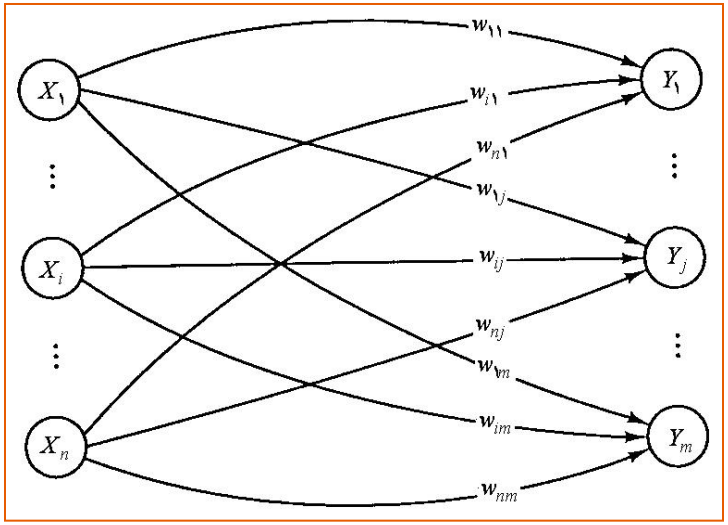
- ایجاد پیوندهایی بین الگوها = یادگیری
  - حافظهٔ انسان مواردی مانند افکار، احساسات و ... را که شبیه، متضاد، مجاور یا با توالی نزدیکی اتفاق می‌افتند، به هم پیوند می‌دهد
  - مثال
    - ارتباط بین عکس یک فرد و خود او
    - ارتباط بین یک نت موسیقی و آهنگ آن
  - ذخیره‌سازی و بازیابی داده‌ها براساس محتوا و نه بر اساس آدرس ذخیره‌سازی
- شبکه‌های حافظهٔ انجمنی (Associative Memory)
  - مدل ساده شده‌ای از حافظهٔ انسان
  - شبکه‌هایی برای ذخیره‌سازی الگوها و پیوند بین آنها

شبکه‌های انجمنی . . .

- شبکه‌های انجمنی
  - یادگیری پیوند یک جفت بردار ورودی—خروجی s:t
- شبکه حافظه خود انجمنی (Auto-Associative Memory)
  - بردارهای ورودی و خروجی یکسان هستند.
- شبکه حافظهٔ دیگرانجمنی (Hetero-Associative Memory)
  - بردارهای ورودی و خروجی با هم متفاوت هستند.
- در هر دو نوع شبکه
  - جفت الگوهای آموزش یافته را یاد می‌گیرد
  - هنگامی‌که یک الگوی ورودی به آن داده می‌شود که مشابه، اما نه یکسان با، ورودی‌های آموزش است، الگوی پاسخ مورد نظر را بازخوانی می‌کند.

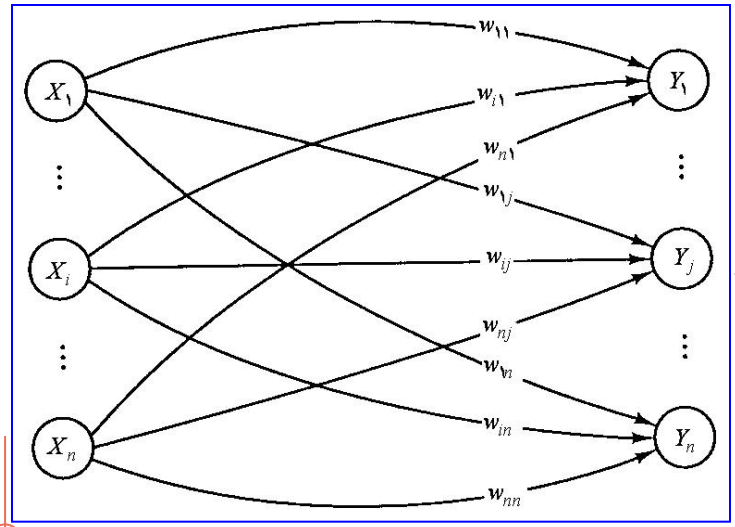
شبکه‌های انجمنی: ساختار . . .

- شبکه یک لایه
- شبکهٔ عصبی دیگرانجمنی



تعداد n ورودی و m خروجی

- شبکهٔ عصبی خودانجمنی



تعداد n ورودی و n خروجی

الگوریتم‌های آموزش پیوند الگو: قانون هب . . .

- مرحلهٔ ۰- وزن‌ها را مقداردهی اولیه نمایید $w_{ij} = 0 ; (i = 1,...,n ; \ j = 1,...,m)$
- مرحلهٔ ۱- برای هر جفت بردار آموزش ورودی-هدف s:t، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- فعال‌سازی‌های واحدهای ورودی شبکه را برای ورودی آموزش فعلی تعیین کنید $x_i = s_i \ ; \ \ (i = 1,...,n)$
- مرحلهٔ ۳- فعال‌سازی‌های واحدهای خروجی شبکه را با توجه به مقادیر هدف تعیین کنید

$$y_j = t_j ; \ \ (j = 1,...,m)$$

- مرحلهٔ ۴- وزن‌ها را تنظیم کنید

$$w_{ij}(new) = w_{ij}(old) + x_i y_j ; \ (i = 1,...,n ; \ j = 1,...,m)$$

الگوریتم‌های آموزش پیوند الگو: قانون هب . . .

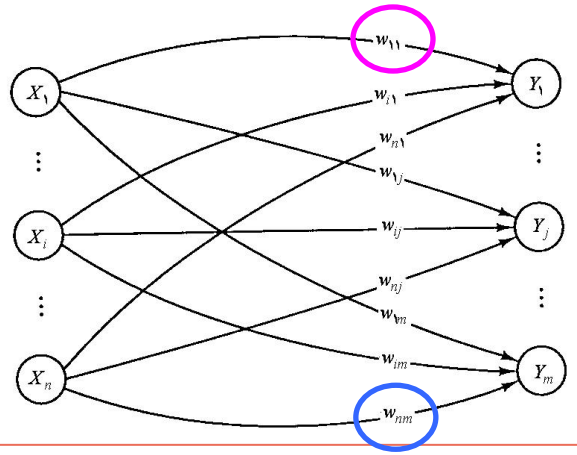
- به‌دست آوردن وزن‌ها قانون هب با ضرب خارجی . . .

ورودی n بعدی

خروجی m بعدی

$$S \times T = \begin{bmatrix} s_1 \\ \vdots \\ s_i \\ \vdots \\ s_n \end{bmatrix} [t_1 \ldots t_j \ldots t_m] = \begin{bmatrix} s_1t_1 & \cdots & s_1t_j & \cdots & s_1t_m \\ \vdots & & \vdots & & \vdots \\ s_it_1 & \cdots & s_it_j & \cdots & s_it_m \\ \vdots & & \vdots & & \vdots \\ s_nt_1 & \cdots & s_nt_j & \cdots & s_nt_m \end{bmatrix}$$

ماتریس وزن ذخیره‌سازی پیوند s:t با روش هب



الگوریتم‌های آموزش پیوند الگو: قانون هب . . .

- به‌دست آوردن وزن‌ها قانون هب با ضرب خارجی . . .
  - قانون یادگیری هب

$$w_{ij}(new) = w_{ij}(old) + x_i y_j \; ; \; (i = 1,\ldots,n \; ; \; j = 1,\ldots,m)$$

  - مقدار اولیهٔ صفر برای وزن‌ها ⇦ وزن‌های نهایی = جمع کلیه $x_i y_j$ها به ازای کلیه بردارهای ورودی–خروجی مختلف

P=تعداد نمونه‌های آموزشی

$$w_{ij} = \sum_{p=1}^{P} s_i(p)\, t_j(p)$$

$$\mathbf{W} = \sum_{p=1}^{P} \mathbf{s}^T(p)\,\mathbf{t}(p)$$

حاصل‌جمع ضرب خارجی ماتریس‌های لازم برای ذخیره‌سازی هر پیوند

الگوریتم‌های آموزش پیوند الگو: قانون هب . . .

- بازیابی . . .
  - کامل شدن آموزش شبکه = ماتریس وزن = دخیره مجموعه‌ای از بردارهای ورودی—هدف
  - تابع همانی برای تابع فعال‌سازی خروجی
  - پاسخ شبکه به بردار ورودی $\mathbf{x}$ برابر $\mathbf{y}=\mathbf{x}\mathbf{W}$

  - سیگنال ورودی، بردار ورودی آموزش kام باشد، یعنی $\mathbf{x}=\mathbf{s}(k)$، آنگاه:
  - باید خروجی شبکه $\mathbf{t}(k)$ باشد

$$\mathbf{s}(k)\mathbf{W}=\sum_{p=1}^{P}\mathbf{s}(k)\mathbf{s}(p)\mathbf{t}(p)=\mathbf{s}(k)\mathbf{s}^{T}(k)\mathbf{t}(k)+\sum_{p\neq k}\mathbf{s}(k)\mathbf{s}^{T}(p)\mathbf{t}(p)$$

مجذور نُرم بردار ورودی= یک عدد نرمال‌سازی

بازیابی کامل = باید برابر صفر باشند

نویز (پارازیت)

الگوریتم‌های آموزش پیوند الگو: قانون هب

بازیابی

$$\sum_{p \neq k} \mathbf{s}(k)\mathbf{s}^T(p)\mathbf{t}(p) = 0 \quad \Rightarrow \quad \mathbf{s}(k)\,\mathbf{s}^T(p) = 0$$

حاصل‌ضرب داخلی = صفر

- صفر بودن حاصل‌ضرب داخلی دو بردار = متعامد (Orthogonal) بودن دو بردار

بازیابی کامل در آموزش با هب

- همبستگی (Correlation) بین بردارهای آموزش ورودی صفر باشد
- متعامد (Orthogonal) بودن بردارهای آموزش ورودی

اگر s(k) با برخی از بردارهای آموزش متعامد نباشد

- پاسخ شبکه به این بردار، شامل مقادیر هدف خود و مقادیر هدف کلیه بردارهای آموزشی که با s(k) همبستگی دارند، است

$$\mathbf{s}(k)\mathbf{W} = \mathbf{s}(k)\mathbf{s}^T(k)\mathbf{t}(p) + \sum_{p \neq k} \mathbf{s}(k)\mathbf{s}^T(p)\mathbf{t}(p)$$

الگوریتم‌های آموزش پیوند الگو: قانون دلتا . . .

قانون دلتای اصلی

- تابع فعال‌سازی برای واحدهای خروجی = تابع همانی
- حداقل کردن مجذور اختلاف بین ورودی شبکه به واحدهای خروجی و مقادیر هدف

$$w_{ij}(new) = w_{ij}(old) + \alpha(t_j - y_j)x_i \; ; \quad (i = 1,....,n; j = 1,....,m)$$

$$y_j = \sum_i x_i w_{ij} \qquad \Delta w_{ij} = \alpha(t_j - y_j)x_i$$

قانون دلتای گسترش‌یافته (Extended Delta Rule)

- تابع فعال‌سازی اختیاری و مشتق‌پذیر برای واحدهای خروجی (به جای تابع همانی)
- متفاوت با قانون دلتا تعمیم‌یافته(Generalized Delta Rule)- الگوریتم پس‌انتشار در آموزش شبکه‌های پرسپترون چندلایه

$$\Delta w_{IJ} = \alpha(t_J - y_J)\, x_I\, f'(y\_in_J)$$

الگوریتم‌های آموزش پیوند الگو: قانون دلتا . . .

استخراج قانون دلتای گسترش‌یافته

- هدف: کمینه کردن مربعات خطا
- تابع خطا= تابعی از تمام وزن‌ها
- راه حل: مشتق جزیی نسبت به وزن‌ها

$$E = \sum_{j=1}^{m} (t_j - y_j)^2$$

$$\frac{\partial E}{\partial w_{IJ}} = \frac{\partial}{\partial w_{IJ}} \sum_{j=1}^{m} (t_j - y_j)^2 = \frac{\partial}{\partial w_{IJ}} (t_J - y_J)^2$$

$$y\_in_J = \sum_{i=1}^{n} x_i w_{iJ} \;\; ; \;\; y_J = f(y\_in_J)$$

$$\frac{\partial E}{\partial w_{IJ}} = -2(t_J - y_J) \frac{\partial y\_in_J}{\partial w_{IJ}} = -2(t_J - y_J) x_I f'(y\_in_J)$$

$$\Delta w_{IJ} = \alpha (t_J - y_J) x_I f'(y\_in_J)$$

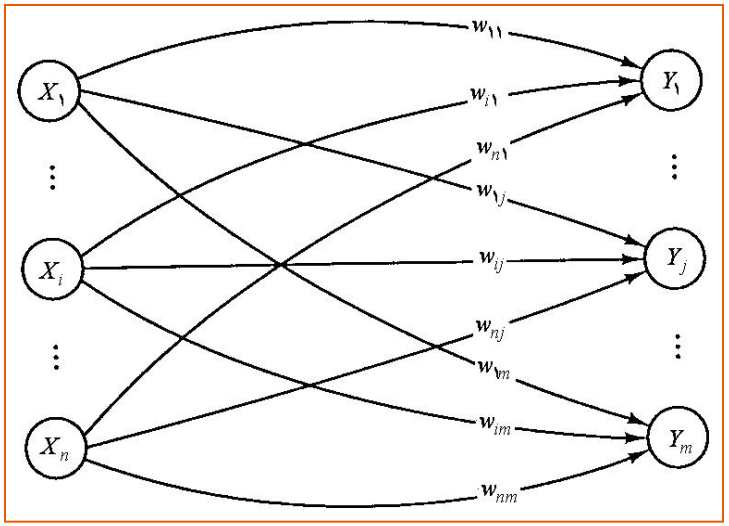
الگوریتم‌های آموزش پیوند الگو: قانون دلتا

- یادگیری الگوهای که به طور خطی مستقل هستند، اما متعامد نیستند
- اجتناب از بروز مشکلات ناشی از نویز
- تولید پاسخی با حداقل مربعات خطا
  - حتی برای شرایطی که الگوهای ورودی به طور خطی مستقل نیستند

- بهتر از قانون هب

شبکهٔ عصبی حافظهٔ دیگرانجمنی . . .

- الگوهای ورودی و خروجی با هم متفاوتند
- آموزش (تنظیم وزن‌ها)
  - قانون هب
  - قانون دلتا
- ساختار



تعداد n ورودی و m خروجی

شبکهٔ عصبی حافظهٔ دیگرانجمنی . . .

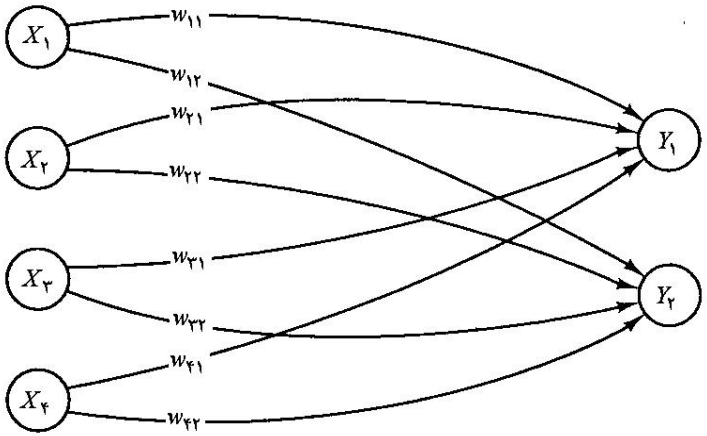
کاربرد

- مرحلهٔ ۰ – با استفاده از قانون هب یا قانون دلتا مقادیر وزن‌ها را به دست آورید.
- مرحلهٔ ۱ – برای هر بردار ورودی، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲– فعال‌سازی‌های واحدهای لایهٔ ورودی شبکه را برابر با بردار ورودی فعلی x_i قرار دهید.
- مرحلهٔ ۳– ورودی شبکه به واحدهای خروجی را محاسبه کنید. $y\_in_j = \sum_j x_i\, w_{ij}$
- مرحلهٔ ۴: فعال‌سازی واحدهای خروجی را مشخص کنید (در اینجا برای مقادیر هدف با نمایش دوقطبی)

$$y_j = \begin{cases} 1 & if \quad y\_in_j > 0 \\ 0 & if \quad y\_in_j = 0 \\ -1 & if \quad y\_in_j < 0 \end{cases}$$

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .



		s_1	s_2	s_3	s_4		t_1	t_2
1	**s**	(1,	0,	0,	0)	**t**	(1,	0)
2	**s**	(1,	1,	0,	0)	**t**	(1,	0)
3	**s**	(0,	0,	0,	1)	**t**	(0,	1)
4	**s**	(0,	0,	1,	1)	**t**	(0,	1)

- بردارهای ورودی با هم متعامد نیستند
- چون مقادیر هدف به صورت ساده‌ای با بردارهای ورودی مرتبط هستند، نویز بین اولین و دومین بردار ورودی مشکلی بوجود نمی‌آورد چون مقادیر هدف آنها یکی است

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب

$$w_{ij}(new) = w_{ij}(old) + s_i\, t_j \;; \quad i.e.,\ \Delta w_{ij} = s_i\, t_j$$

- مرحلهٔ ۰ – به تمام وزن‌ها مقدار اولیهٔ صفر بدهید.
- مرحلهٔ ۱ – برای اولین جفت ورودی–خروجی، $\mathbf{s:t} \Rightarrow (1, 0, 0, 0):(1, 0)$
- مرحلهٔ ۲ – $x_1 = 1,\ x_2 = x_3 = x_4 = 0$
- مرحلهٔ ۳ – $y_1 = 1,\ y_2 = 0$
- مرحلهٔ ۴ – $w_{11}(new) = w_{11}(old) + x_1 y_1 = 0 + 1 = 1$

(سایر وزن‌ها برابر صفر باقی می‌مانند)

- مرحلهٔ ۱ – برای دومین جفت ورودی–خروجی، $\mathbf{s:t} \Rightarrow (1, 1, 0, 0):(1, 0)$
- مرحلهٔ ۲ – $x_1 = x_2 = 1,\ x_3 = x_4 = 0$
- مرحلهٔ ۳ – $y_1 = 1,\ y_2 = 0$
- مرحلهٔ ۴ – $w_{21}(new) = w_{21}(old) + x_2 y_1 = 0 + 1 = 1$ $\quad w_{11}(new) = w_{11}(old) + x_1 y_1 = 1 + 1 = 2$

(سایر وزن‌ها برابر صفر باقی می‌مانند)

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

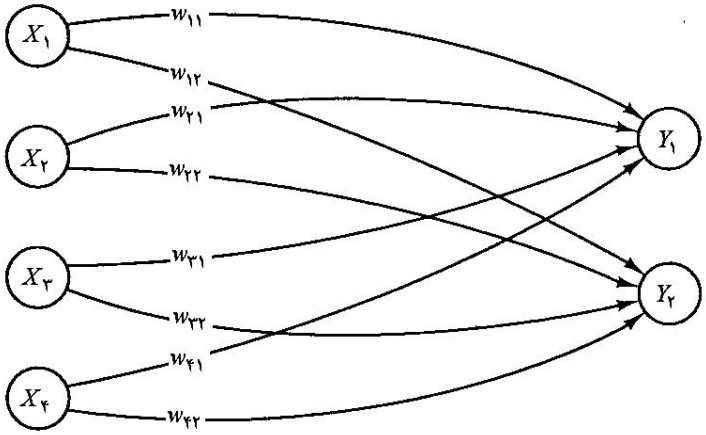
- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب
  - مرحلهٔ ۱- برای سومین جفت ورودی-خروجی، $\mathbf{s:t} \Rightarrow (0,0,0,1):(0,1)$
  - مرحلهٔ ۲- $x_4 = 1,\ x_1 = x_2 = x_3 = 0$
  - مرحلهٔ ۳- $y_1 = 0,\ y_2 = 1$
  - مرحلهٔ ۴- $w_{42}(new) = w_{42}(old) + x_4 y_2 = 0+1=1$

    (سایر وزن‌ها برابر صفر باقی می‌مانند)
  - مرحلهٔ ۱- برای چهارمین جفت ورودی-خروجی، $\mathbf{s:t} \Rightarrow (0,0,1,1):(0,1)$
  - مرحلهٔ ۲- $x_1 = x_2 = 0,\ x_3 = x_4 = 1$
  - مرحلهٔ ۳- $y_1 = 0,\ y_2 = 1$
  - مرحلهٔ ۴- $w_{32}(new) = w_{32}(old) + x_3 y_2 = 0+1=1$ $\quad w_{42}(new) = w_{42}(old) + x_4 y_2 = 1+1=2$

    (سایر وزن‌ها برابر صفر باقی می‌مانند)

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب



		s_1	s_2	s_3	s_4		t_1	t_2
1	**s**	(1,	0,	0,	0)	**t**	(1,	0)
2	**s**	(1,	1,	0,	0)	**t**	(1,	0)
3	**s**	(0,	0,	0,	1)	**t**	(0,	1)
4	**s**	(0,	0,	1,	1)	**t**	(0,	1)

- وزن‌های نهایی

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

جمع حاصل‌ضرب s(1)ها در t(1)های متناسب

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ضرب‌های خارجی
  - اولین جفت الگو $\mathbf{s:t} \Rightarrow (1, 0, 0, 0):(1, 0)$

$$\begin{bmatrix}1\\0\\0\\0\end{bmatrix}.\begin{bmatrix}1 & 0\end{bmatrix}=\begin{bmatrix}1 & 0\\0 & 0\\0 & 0\\0 & 0\end{bmatrix}$$

  - دومین جفت الگو $\mathbf{s:t} \Rightarrow (1, 1, 0, 0):(1, 0)$

$$\begin{bmatrix}1\\1\\0\\0\end{bmatrix}.\begin{bmatrix}1 & 0\end{bmatrix}=\begin{bmatrix}1 & 0\\1 & 0\\0 & 0\\0 & 0\end{bmatrix}$$

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ضرب‌های خارجی
  - سومین جفت الگو $\mathbf{s:t} \Rightarrow (0,0,0,1):(0,1)$

$$\begin{bmatrix}0\\0\\0\\1\end{bmatrix}.\begin{bmatrix}0 & 1\end{bmatrix}=\begin{bmatrix}0 & 0\\0 & 0\\0 & 0\\0 & 1\end{bmatrix}$$

  - چهارمین جفت الگو $\mathbf{s:t} \Rightarrow (0,0,1,1):(0,1)$

$$\begin{bmatrix}0\\0\\1\\1\end{bmatrix}.\begin{bmatrix}0 & 1\end{bmatrix}=\begin{bmatrix}0 & 0\\0 & 0\\0 & 1\\0 & 1\end{bmatrix}$$

- ماتریس وزن نهایی = مجموع ماتریس‌های وزن هر کدام از جفت الگوها

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

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ارزیابی (داده‌های آموزش به عنوان ورودی)

$$f(x)=\begin{cases}1 \ if \ \ x>0\\ 0 \ if \ \ x\le 0\end{cases}$$

- مرحلهٔ ۰- $\mathbf{W}=\begin{bmatrix}2 & 0\\ 1 & 0\\ 0 & 1\\ 0 & 2\end{bmatrix}$
- مرحلهٔ ۱- برای اولین الگوی ورودی، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- $\mathbf{x}=(1,\ 0,0,\ 0)$
- مرحلهٔ ۳- $y\_in_1=x_1w_{11}+x_2w_{21}+x_3w_{31}+x_4w_{41}=1\times 2+0\times 1+0\times 0+0\times 0=2$

$$y\_in_2=x_1w_{12}+x_2w_{22}+x_3w_{32}+x_4w_{42}=1\times 0+0\times 0+0\times 1+0\times 2=0$$

- مرحلهٔ ۴- $y_1=f\ (y\_in_1)=f\ (2)=1$

$$y_2=f\ (y\_in_2)=f\ (0)=0$$

پاسخ صحیح برای اولین الگوی آموزش

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ارزیابی (داده‌های آموزش به عنوان ورودی)

$$f(x)=\begin{cases}1 \ if \ \ x>0\\ 0 \ if \ \ x\le 0\end{cases}$$

- مرحلهٔ ۰- $\mathbf{W}=\begin{bmatrix}2 & 0\\ 1 & 0\\ 0 & 1\\ 0 & 2\end{bmatrix}$
- مرحلهٔ ۱- برای دومین الگوی ورودی، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- $\mathbf{x}=(1,\ 1,\ 0,\ 0)$
- مرحلهٔ ۳- $y\_in_1=x_1w_{11}+x_2w_{21}+x_3w_{31}+x_4w_{41}=1\times 2+1\times 1+0\times 0+0\times 0=3$

$$y\_in_2=x_1w_{12}+x_2w_{22}+x_3w_{32}+x_4w_{42}=1\times 0+0\times 0+0\times 1+0\times 2=0$$

- مرحلهٔ ۴- $y_1=f\ (y\_in_1)=f\ (3)=1$

$$y_2=f\ (y\_in_2)=f\ (0)=0$$

پاسخ صحیح برای دومین الگوی آموزش

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ارزیابی (داده‌های آموزش به عنوان ورودی)

$$f(x)=\begin{cases}1 \ if \ \ x>0\\ 0 \ if \ \ x\le 0\end{cases}$$

- مرحلهٔ ۰- $\mathbf{W}=\begin{bmatrix}2 & 0\\ 1 & 0\\ 0 & 1\\ 0 & 2\end{bmatrix}$
- مرحلهٔ ۱- برای سومین الگوی ورودی، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- $\mathbf{x}=(0,\ 0,\ 0,\ 1)$
- مرحلهٔ ۳- $y\_in_1=x_1w_{11}+x_2w_{21}+x_3w_{31}+x_4w_{41}=0\times 2+0\times 1+0\times 0+1\times 0=0$

$$y\_in_2=x_1w_{12}+x_2w_{22}+x_3w_{32}+x_4w_{42}=0\times 0+0\times 0+0\times 1+1\times 2=2$$

- مرحلهٔ ۴- $y_1=f\ (y\_in_1)=f\ (0)=0$

$$y_2=f\ (y\_in_2)=f\ (2)=1$$

پاسخ صحیح برای سومین الگوی آموزش

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ارزیابی (داده‌های آموزش به عنوان ورودی)

$$f(x)=\begin{cases}1 \ if \ \ x>0\\ 0 \ if \ \ x\le 0\end{cases}$$

- مرحلهٔ ۰- $\mathbf{W}=\begin{bmatrix}2 & 0\\ 1 & 0\\ 0 & 1\\ 0 & 2\end{bmatrix}$
- مرحلهٔ ۱- برای چهارمین الگوی ورودی، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- $\mathbf{x}=(0,\ 0,1,\ 1)$
- مرحلهٔ ۳- $y\_in_1=x_1w_{11}+x_2w_{21}+x_3w_{31}+x_4w_{41}=0\times 2+0\times 1+1\times 0+1\times 0=0$

$$y\_in_2=x_1w_{12}+x_2w_{22}+x_3w_{32}+x_4w_{42}=0\times 0+0\times 0+1\times 1+1\times 2=3$$

- مرحلهٔ ۴- $y_1=f\ (y\_in_1)=f\ (0)=0$

$$y_2=f\ (y\_in_2)=f\ (3)=1$$

پاسخ صحیح برای چهارمین الگوی آموزش

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- بردارهای ورودی چهار بعدی و بردارهای خروجی دو بعدی . . .
- آموزش با قانون هب: ارزیابی (داده‌های غیرآموزش یافته)
  - مرحلهٔ ۰- $\mathbf{W} = \begin{bmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{bmatrix}$
  - مرحلهٔ ۱- برای بردار ورودی $\mathbf{x} = (0,\ 1, 0,\ 0)$

$$(0,1,0,0).\mathbf{W} = (1,0) \rightarrow (1,0)$$

شبیه به الگوی آموزش یافته

$\mathbf{s} = (1,\ 1, 0,\ 0)$

پیوند الگوی خروجی شناخته شده $(1,\ 0)$ با الگوی ورودی

  - مرحلهٔ ۱- برای بردار ورودی $(0,\ 1, 1,\ 0)$

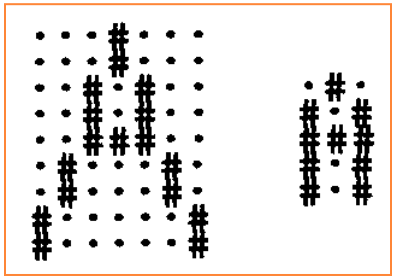
$$(0,1,1,0).\mathbf{W} = (1,1) \rightarrow (1,1)$$

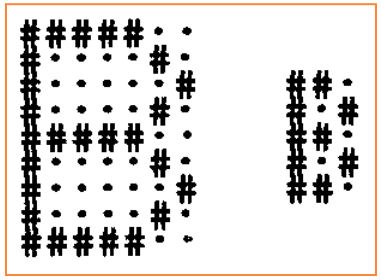
تفاوت در دو مؤلفه با الگوهای آموزش

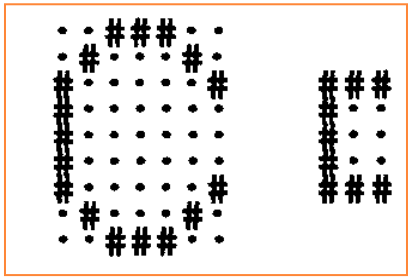
عدم پیوند الگوی درست با الگوهای خروجی شناخته شده

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- شبکهٔ دیگرانجمنی برای پیوند دادن حروفی با فونت‌های مختلف . . .
  - آموزش یک شبکهٔ عصبی دیگرانجمنی با قانون هب (ضرب خارجی)
  - پیوند سه جفت بردار حروف A، B و C: پیوند حروف با اندازه بزرگ به اندازه کوچک
  - ورودی شبکه = بردارهای ۶۳ مؤلفه‌ای (حروف با اندازه بزرگ)
  - خروجی شبکه = بردارهای ۱۵ مؤلفه‌ای (حروف با اندازه کوچک)







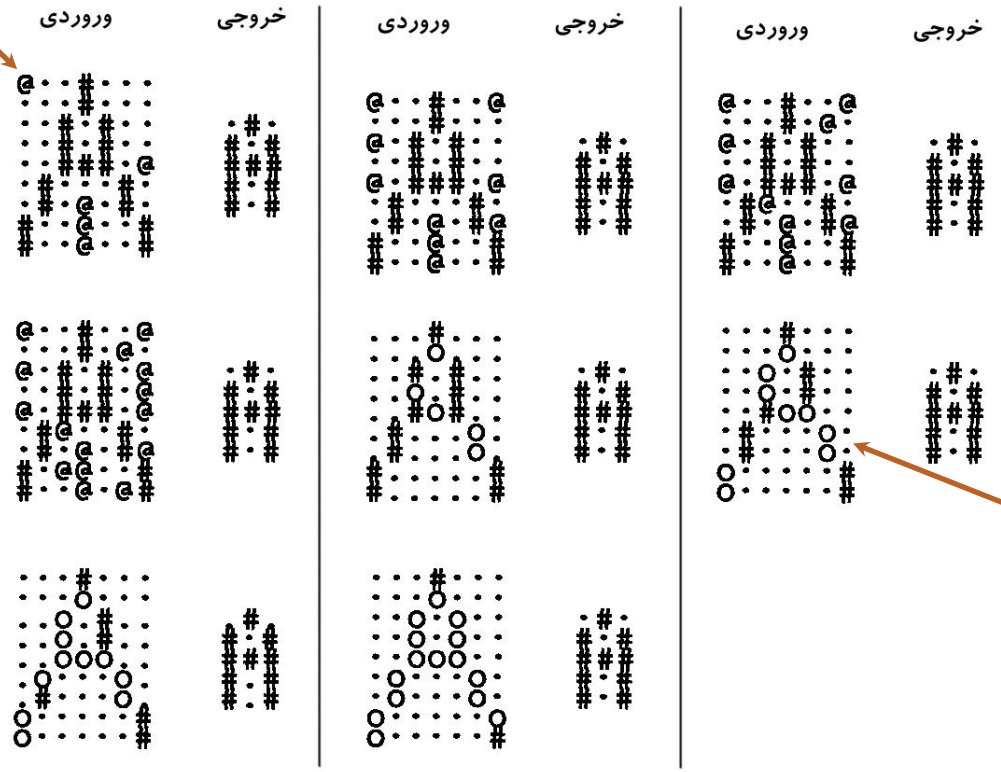
- نمایش: # = ۱ و نقطه = ۱- ؛ خواندن ردیف به ردیف با شروع از ردیف بالا

$$(-1,\ 1,-1,\quad 1,-1,1,\quad 1,1,1,\quad 1,-1,1,\quad 1,-1,1)$$

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال . . .

- شبکهٔ دیگرانجمنی برای پیوند دادن حروفی با فونت‌های مختلف . . .
  - آموزش با قانون هب (ضرب خارجی)
  - ارزیابی با حالت نویزی الگوهای ورودی

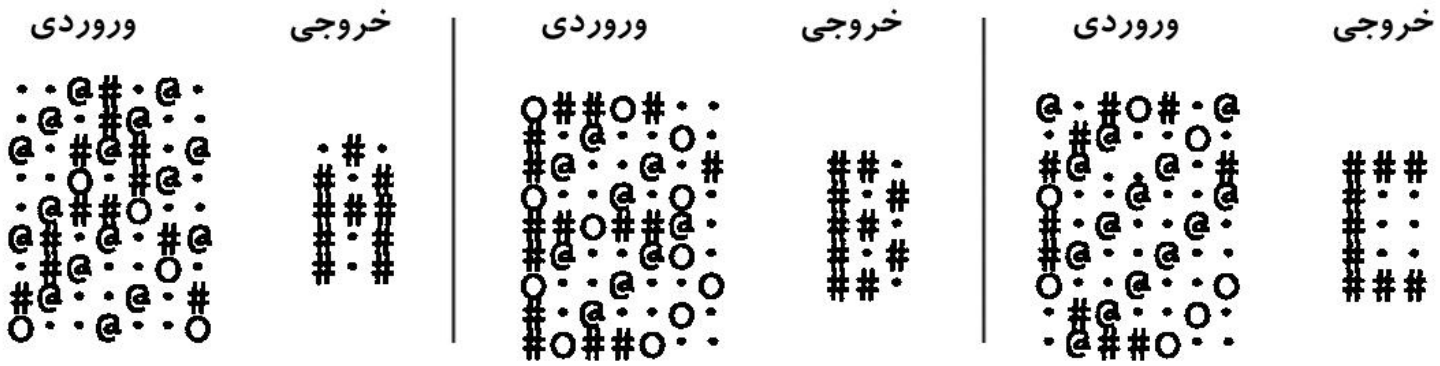
@ = پیکسل اکنون «فعال» است، باید «غیرفعال» می‌بود (نویز)



O = پیکسل اکنون «غیرفعال» است، اما باید «فعال» می‌بود (نویز)

شبکهٔ عصبی حافظهٔ دیگرانجمنی: مثال

- شبکهٔ دیگرانجمنی برای پیوند دادن حروفی با فونت‌های مختلف
  - آموزش با قانون هب (ضرب خارجی)
  - ارزیابی با حالت نویزی الگوهای ورودی
  - اشتباهاتی در یک-سوم مؤلفه‌ها

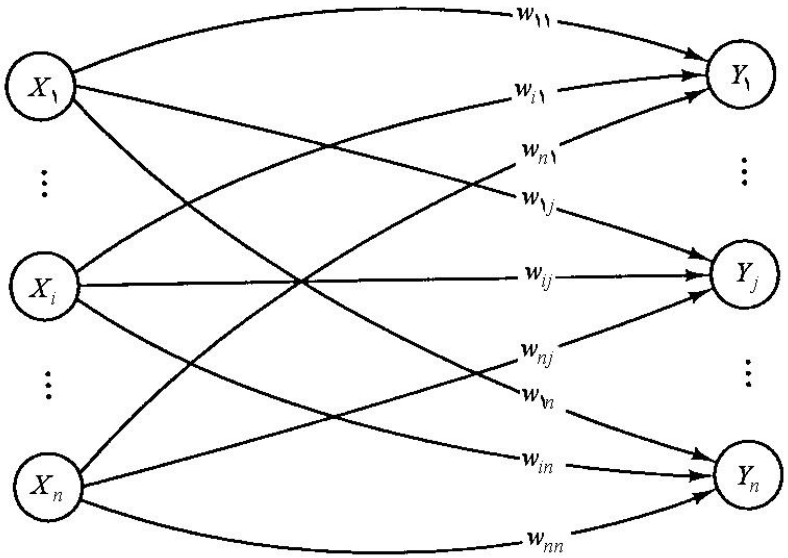


شناسایی درست برای
ورودی‌های با ۳۰٪ نویز

شبکهٔ خودانجمنی . . .

- بردارهای ورودی آموزش و خروجی هدف یکسان هستند
- آموزش در این شبکه‌ها = ذخیره‌سازی بردارها

- بازیابی بردار ورودی (حتی در حالت خراب شدن یا نویزی شدن) در صورت شبیه بودن به یکی از بردارهای ذخیره شده



- ساختار
  - حالتی خاص از شبکهٔ دیگرانجمنی

شبکهٔ خودانجمنی: الگوریتم . . .

- مرحلهٔ ۰- به تمام وزن‌ها مقدار اولیه بدهید، $w_{ij}=0,\ i=1,\ldots,n;\ j=1,\ldots,m$
- مرحلهٔ ۱- برای هر برداری که می‌خواهیم ذخیره کنیم، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- فعال‌سازی هر واحد ورودی را تعیین کنید، $x_i=s_i,\ i=1,\ldots,n$

$n=m$

- مرحلهٔ ۳- فعال‌سازی واحدهای خروجی را تعیین کنید، $y_j=s_j,\ j=1,\ldots,m$
- مرحلهٔ ۴- وزن‌ها را به صورت زیر تنظیم کنید،

$$w_{ij}(new)=w_{ij}(old)+x_i y_j \qquad i=1,\ldots,n;\ j=1,\ldots,m$$

یادگیری با قانون هب

$$\mathbf{W}=\sum_{p=1}^{P} s^T(p)s(p)$$

یادگیری بردارهای متعامد

H. Veisi (h.veisi@ut.ac.ir)

شبکهٔ خودانجمنی: کاربرد . . .

- مرحلهٔ ۰- وزن‌های شبکه را مشخص کنید (با استفاده از قانون هب یا همان ضرب خارجی)
- مرحلهٔ ۱- برای هر بردار ورودی آزمایش، مراحل ۲ تا ۴ را انجام دهید.
- مرحلهٔ ۲- فعال‌سازی‌های واحدهای ورودی را برابر با بردار ورودی قرار دهید.
- مرحلهٔ ۳- ورودی شبکه به هر واحد خروجی را محاسبه کنید،
- $y\_in_j = \sum_i x_i w_{ij} \quad j = 1, \ldots, m$
- مرحلهٔ ۴- تابع فعال‌سازی زیر را به ازای $j = 1, \ldots, m$ به کار ببرید

$$y_j = f(y\_in_j) = \begin{cases} 1 \ if \quad y\_in_j > 0 \\ -1 \ if \quad y\_in_j \le 0 \end{cases}$$

بردار ورودی شبکه

- «شناخته شده» = ذخیره شده در شبکه
  - تولید الگوی فعال‌سازی در واحدهای خروجی (همانند یکی از بردارهای ذخیره شده است)
- «ناشناس»

شبکهٔ خودانجمنی: مثال . . .

- مرحلهٔ ۰- بردار $\mathbf{s}=(1,1,1,-1)$ با ماتریس وزن زیر ذخیره شده است:

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

- مرحلهٔ ۱- برای بردار ورودی آزمایش:
- مرحلهٔ ۲- $\mathbf{x}=(1,1,1,-1)$
- مرحلهٔ ۳- $\mathbf{y}\_in=(4,4,4,-4)$
- مرحلهٔ ۴- $\mathbf{y}=f(4,4,4,-4)=(1,1,1,-1)$

$$(1,\ 1,\ 1,\ -1).\mathbf{W}=(4,\ 4,\ 4,\ -4)\rightarrow(1,\ 1,\ 1,\ -1)$$

شناسایی بردار ورودی به عنوان بردار «شناخته شده»

شبکهٔ خودانجمنی: مثال . . .

- بررسی شبکه با داده‌های «اشتباه» یا داده‌های «گم‌شده»

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

- مرحلهٔ ۰ – بردار $\mathbf{s} = (1, 1, 1, -1)$ با ماتریس وزن
- مرحلهٔ ۱ – برای بردار ورودی با یک مولفه «اشتباه»

$$(-1 \quad 1, \quad 1, \quad -1) \cdot \mathbf{W} = (2, \quad 2, \quad 2, \quad -2) \rightarrow (1, \quad 1, \quad 1, \quad -1)$$
$$(\ 1 \quad -1, \quad 1, \quad -1) \cdot \mathbf{W} = (2, \quad 2, \quad 2, \quad -2) \rightarrow (1, \quad 1, \quad 1, \quad -1)$$
$$(\ 1 \quad 1, \quad -1, \quad -1) \cdot \mathbf{W} = (2, \quad 2, \quad 2, \quad -2) \rightarrow (1, \quad 1, \quad 1, \quad -1)$$
$$(\ 1 \quad 1, \quad 1, \quad 1) \cdot \mathbf{W} = (2, \quad 2, \quad 2, \quad -2) \rightarrow (1, \quad 1, \quad 1, \quad -1)$$

شناسایی درست

یک اشتباه در بردار ورودی

- مرحلهٔ ۱ – برای بردار ورودی با یک مولفه «گم‌شده»
  - شناسایی درست : (0,1,1,-1)، (1,0,1,-1)، (1,1,0,-1) و (1,1,1,0)

H. Veisi (h.veisi@ut.ac.ir)

شبکهٔ خودانجمنی: مثال . . .

- بررسی شبکه با **دو** مولفه «گم‌شده» در بردار ورودی

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

- مرحلهٔ ۰ – بردار $\mathbf{s} = (1, 1, 1, -1)$ با ماتریس وزن
- مرحلهٔ ۱ – برای بردار ورودی با **دو** مولفه «گم‌شده»

$$\begin{aligned}
(0,\ 0,\ 1,\ -1)\ .\ \mathbf{W} &= (2,\ 2,\ 2,\ -2) \rightarrow (1,\ 1,\ 1,\ -1) \\
(0,\ 1,\ 0,\ -1)\ .\ \mathbf{W} &= (2,\ 2,\ 2,\ -2) \rightarrow (1,\ 1,\ 1,\ -1) \\
(0,\ 1,\ 1,\ 0)\ .\ \mathbf{W} &= (2,\ 2,\ 2,\ -2) \rightarrow (1,\ 1,\ 1,\ -1) \\
(1,\ 0,\ 0,\ -1)\ .\ \mathbf{W} &= (2,\ 2,\ 2,\ -2) \rightarrow (1,\ 1,\ 1,\ -1) \\
(1,\ 0,\ 1,\ 0)\ .\ \mathbf{W} &= (2,\ 2,\ 2,\ -2) \rightarrow (1,\ 1,\ 1,\ -1) \\
(1,\ 1,\ 0,\ 0)\ .\ \mathbf{W} &= (2,\ 2,\ 2,\ -2) \rightarrow (1,\ 1,\ 1,\ -1)
\end{aligned}$$

شناسایی درست

دو مولفه گم‌شده در بردار ورودی

شبکهٔ خودانجمنی: مثال

- بررسی شبکه با دو مولفه «اشتباه» در بردار ورودی

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

- مرحلهٔ ۰- بردار $\mathbf{s}=(1,1,1,-1)$ با ماتریس وزن
- مرحلهٔ ۱- برای بردار ورودی با دو مولفه «اشتباه»

$$(-1,\ -1,\ 1,\ -1).\mathbf{W}=(0,\ 0,\ 0,\ 0)$$

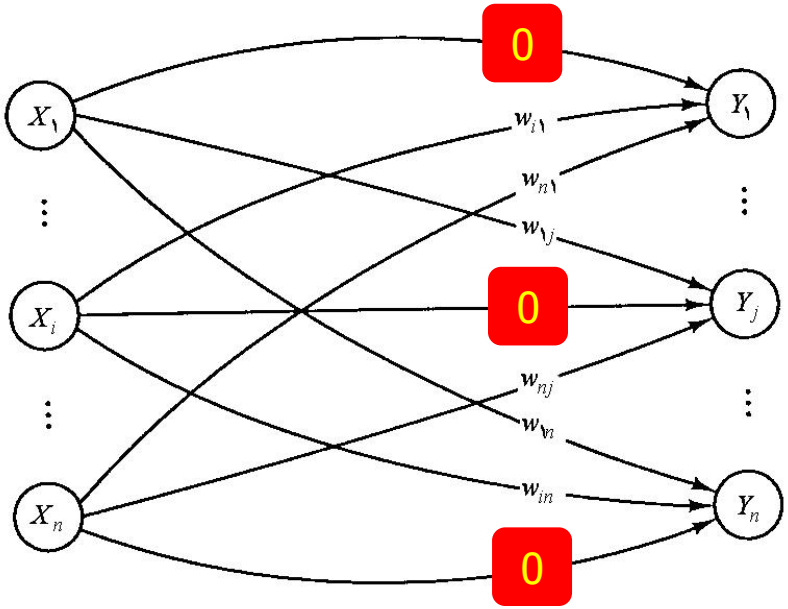
شناسایی نادرست

- جمع‌بندی
  - شبکه در برابر داده‌های «گم‌شده» مقاوم‌تر از «اشتباه» در داده‌ها است
  - بردارهای حاوی داده‌های «گم‌شده» بیشتر از بردارهای حاوی «اشتباه» به الگوهای آموزش نزدیک‌ترند (هم از نظر ظاهری و هم از نظر ریاضیاتی)

شبکهٔ خودانجمنی: بهبود . . .

- شبکهٔ خودانجمنی بدون اتصال سرخود (Self-Connection)
  - صفر کردن مقادیر قطر اصلی در ماتریس وزن

$$\mathbf{W}_0 = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$



شبکهٔ خودانجمنی: بهبود

- شبکهٔ خودانجمنی بدون اتصال سرخود (Self-Connection)

$$\mathbf{W}_0 = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

- بردار ورودی $\mathbf{s} = (1, 1, 1, -1)$
- دو مولفه «اشتباه»

$$(-1, -1, 1, -1).\mathbf{W}_0 = (-1, 1, -1, 1)$$

شناسایی نادرست

- دو مولفه «گم‌شده»

$$\begin{aligned}
(0, 0, 1, -1) . \mathbf{W}_0 &= (2, 2, 1, -1) \rightarrow (1, 1, 1, -1) \\
(0, 1, 0, -1) . \mathbf{W}_0 &= (2, 1, 2, -1) \rightarrow (1, 1, 1, -1) \\
(0, 1, 1, 0) . \mathbf{W}_0 &= (2, 1, 1, -2) \rightarrow (1, 1, 1, -1) \\
(1, 0, 0, -1) . \mathbf{W}_0 &= (1, 2, 2, -1) \rightarrow (1, 1, 1, -1) \\
(1, 0, 1, 0) . \mathbf{W}_0 &= (1, 2, 1, -2) \rightarrow (1, 1, 1, -1) \\
(1, 1, 0, 0) . \mathbf{W}_0 &= (1, 1, 2, -2) \rightarrow (1, 1, 1, -1)
\end{aligned}$$

شناسایی درست

شبکهٔ خودانجمن تکراری . . .

- حالت گسترش یافته‌ای از شبکهٔ خودانجمنی
- پاسخ شبکه به یک الگوی ورودی خاص دوباره به عنوان ورودی به شبکه داده می‌شود.
  - در مواردی، شبکه به سیگنال ورودی‌ای با الگوی هدف ذخیره شده فوراً پاسخ صحیح نمی‌دهد، هر چند پاسخ داده شده به اندازهٔ کافی به یک الگوی ذخیره شده شبیه باشد.
  - دادن پاسخ اول به عنوان ورودی مجدد به شبکه، شانس صحیح پاسخ دادن شبکه را افزایش می‌دهد.
  - اثربخشی در مقابله با داده‌های نویزی

شبکهٔ خودانجمن تکراری: مثال . . .

- بردار ذخیره شده‌ای با ۳ مؤلفه گم‌شده از ۴ مولفه
  - بردار آموزشی $(1, 1, 1, -1)$
  - ماتریس مربوطه

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

- بردار وردی شبکه: بردار ذخیره شده با سه مؤلفهٔ «گم‌شده» $(1, 0, 0, 0)$

$$(1,0,0,0).\mathbf{W}=(0,1,1,-1) \quad \rightarrow \quad (0,1,1,-1).\mathbf{W}=(3,2,2,-2)\rightarrow(1,1,1,-1)$$

پاسخ نادرست برای اولین تکرار

پاسخ درست بعد از دومین تکرار

شبکه خودانجمن تکراری خطی . . .

- شبکه خودانجمن خطی بازگشتی (Recurrent Linear Autoassociator)
  - هر نرون به تمام نرون‌های دیگر (و خودش) متصل است
  - تابع فعال‌سازی هر واحد تابع همانی است = شبکه خودانجمن خطی
  - آموزش با قانون هب (ماتریس متقارن وزن‌ها)
- شبکهٔ حالت مغز در جعبه (Brain-State-in-a-Box) . . .
  - تغییر تابع فعال‌سازی همانی شبکه خودانجمن خطی
    - محدود کردن پاسخ شبکه از رشد بی‌رویه
  - محدود کردن مقادیر تابع فعال‌سازی به داخل یک مکعب
    - محدود کردن مقدار هر مؤلفه خروجی شبکه به مقادیر بین ۱- و ۱
- شبکهٔ خودانجمن با تابع آستانه
  - استفاده از تابع آستانه به عنوان تابع فعال‌سازی
  - تعمیم ساده شبکه BSB با تغییر سطوح تصمیم‌گیری تابع فعال‌سازی

شبکهٔ هاپفیلد گسسته ...

- معرفی
  - ارائه توسط هاپفیلد (استاد فیزیک)
  - اوایل دهه ۸۰ میلادی (سال‌های ۱۹۸۲ و ۱۹۸۴)
  - یک شبکهٔ خودانجمن تکراری شبیه به شبکه‌های توصیف شده
  - یک شبکهٔ عصبی کاملاً به هم متصل بوده و دارای وزن‌های متقارن و بدون اتصال به خود

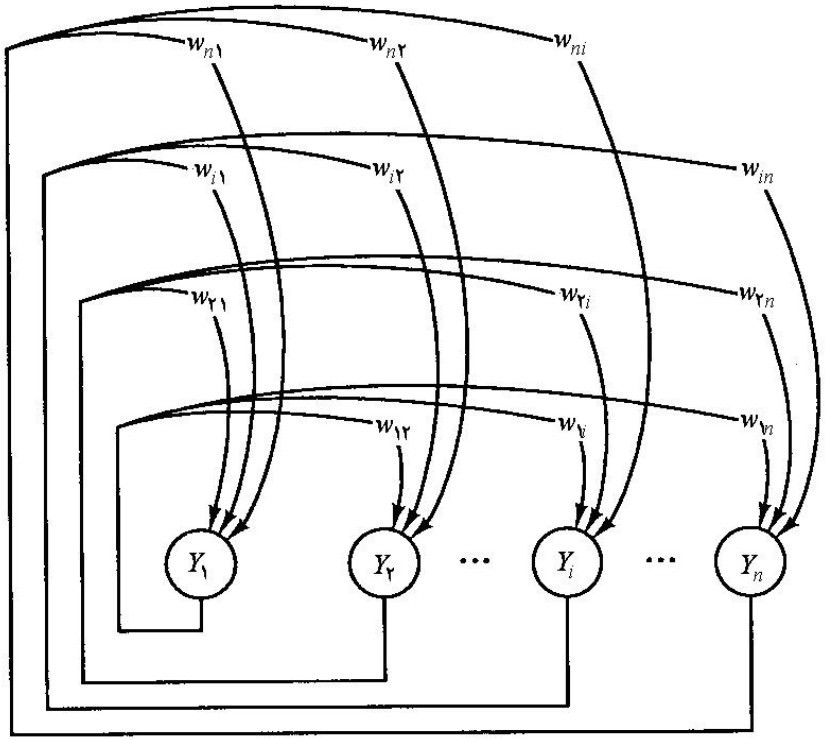
  - دارای تفاوت‌های اندک اما مهم با شبکه‌های دیگر خودانجمن تکراری (تأثیر بر همگرایی)
    - در هر بار فقط یکی از واحدها فعال‌سازی خود را به‌روز می‌کند که این به‌روز کردن بر اساس سیگنال‌های دریافتی از واحدهای دیگر است.
    - هر واحد شبکه، علاوه بر سیگنال‌های دریافتی از سایر واحدهای شبکه، یک سیگنال خارجی را نیز دریافت می‌کند که همان ورودی شبکه است

  - اثبات همگرایی فعال‌سازی‌ها
    - به کمک تابع انرژی یا تابع لیاپونف (Lyapunov Function)

H. Veisi (h.veisi@ut.ac.ir)

شبکهٔ هاپفیلد گسسته: ساختار . . .

- خروجی هر واحد، به همه واحدهای دیگر (به غیر از خودش) به عنوان ورودی با وزن مربوطه وارد می‌شود



شبکهٔ هاپفیلد گسسته: الگوریتم . . .

- آموزش = ذخیره کردن الگوهای ورودی
  - با قانون هب
  - الگوهای ورودی دودویی

$$w_{ij} = \sum_p (2s_i(p)-1)(2s_i(p)-1), \quad i \neq j \; ; \qquad w_{ii} = 0$$

  - الگوهای ورودی دوقطبی

$$w_{ij} = \sum_p s_i(p)s_j(p), \quad i \neq j \quad ; \quad w_{ii} = 0$$

شبکهٔ هاپفیلد گسسته: کاربرد . . .

- مرحلهٔ ۰- الگوها را به صورت وزن‌ها در شبکه ذخیره نمایید (با استفاده از قانون هب).
  تا زمانی‌که فعال‌سازی‌های شبکه همگرا نشده‌اند، مراحل ۱ تا ۷ را انجام دهید.
- مرحلهٔ ۱- برای هر بردار ورودی x، مراحل ۲ تا ۶ را انجام دهید.
- مرحلهٔ ۲- فعال‌سازی‌های اولیهٔ شبکه را برابر با بردار ورودی خارجی x قرار دهید:
- مرحلهٔ ۳- برای هر واحد Y_i، مراحل ۴ تا ۶ را انجام دهید. $y_i = x_i \quad , \quad i = 1,\ldots,n$
  (واحدها به صورت تصادفی به‌روز شوند)
- مرحلهٔ ۴- ورودی شبکه را محاسبه کنید: $y\_in_i = x_i + \sum_j y_j w_{ji}$
- مرحلهٔ ۵- فعال‌سازی شبکه (سیگنال خروجی) را تعیین کنید.

$$y_i = \begin{cases} 1 & if \quad y\_in_i > \theta_i \\ y_i & if \quad y\_in_i = \theta_i \\ 0 & if \quad y\_in_i < \theta_i . \end{cases}$$

- مرحلهٔ ۶- مقدار y_i را به تمام واحدهای دیگر ارسال نمایید.
  (این کار منجر به به‌روز شدن بردار فعال‌سازی یا همان خروجی شبکه می‌شود)
- مرحلهٔ ۷- همگرایی شبکه را بررسی کنید.

شبکهٔ هاپفیلد گسسته: کاربرد . . .

- **چند نکته**
  - مقدار آستانه θ_i ، معمولاً برابر صفر قرار داده می‌شود.
  - ترتیب به‌روز کردن واحدها تصادفی است اما باید میانگین تعداد بارهای به‌روز کردن همه واحدها یکسان باشد
  - در نسخه اولیه شبکهٔ هاپفیلد گسسته
    - استفاده از فعال‌سازی‌های دودویی
    - دادن ورودی خارجی به شبکه فقط در اولین مرحلهٔ زمانی
  - در نسخه‌های بعدی
    - به وجود آمدن امکان دریافت ورودی خارجی در حین انجام فرآیند
    - استفاده از فعال‌سازی‌های دوقطبی

شبکهٔ هاپفیلد گسسته: مثال . . .

- **بردار ذخیره شده‌ای با ۲ مؤلفه اشتباه از ۴ مولفه . . .**
  - **بردار آموزشی** $(1, 1, 1, 0)$ **یا معادل دوقطبی آن** $(1, 1, 1, -1)$
  - **ماتریس مربوطه**

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

  - **ورودی شبکه** $= (0, 0, 1, 0)$
    - دارای اشتباهاتی در مؤلفه اول و دوم در مقایسه با بردار ذخیره شده
  - **به‌روز کردن تصادفی فعال‌سازی‌ها:** Y_1 ، Y_4 ، Y_3 و Y_2

شبکهٔ هاپفیلد گسسته: مثال . . .

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

- بردار ذخیره شده‌ای با ۲ مؤلفه اشتباه از ۴ مولفه . . .
  - مرحلهٔ ۱- بردار ورودی = $(0,0,1,0)$
  - مرحلهٔ ۲- $y=(0,0,1,0)$
  - مرحلهٔ ۳- واحد Y_1 را برای به‌روز کردن فعال‌سازی خود انتخاب کنید.
  - مرحلهٔ ۴- $y\_in_1 = x_1 + \sum_j y_j w_{j1} = 0+1$
  - مرحلهٔ ۵- $y\_in_1 \overset{j}{>} 0 \rightarrow y_1 = 1$
  - مرحلهٔ ۶- $y=(1,0,1,0)$

  - مرحلهٔ ۳- واحد Y_4 را انتخاب کنید تا فعال‌سازی خود را به‌روز کند.
  - مرحلهٔ ۴- $y\_in_4 = x_4 + \sum_j y_j w_{j4} = 0+(-2)$
  - مرحلهٔ ۵- $y\_\overset{j}{in}_4 < 0 \rightarrow y_4 = 0$
  - مرحلهٔ ۶- $y=(1,0,1,0)$

استفاده از فعال‌سازی‌های جدید

شبکهٔ هاپفیلد گسسته: مثال . . .

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

- بردار ذخیره شده‌ای با ۲ مؤلفه اشتباه از ۴ مولفه
  - مرحلهٔ ۶- $y=(1,0,1,0)$
  - مرحلهٔ ۳- واحد Y_3 را برای به‌روز کردن فعال‌سازی خود انتخاب کنید.
  - مرحلهٔ ۴- $y\_in_3=x_3+\sum_j y_j w_{j3}=1+1$
  - مرحلهٔ ۵- $y\_in_3>0 \rightarrow y_3=1$
  - مرحلهٔ ۶- $y=(1,0,1,0)$
  - مرحلهٔ ۳- واحد Y_2 را انتخاب کنید تا فعال‌سازی خود را به‌روز کند.
  - مرحلهٔ ۴- $y\_in_2=x_2+\sum_j y_j w_{j2}=0+2$
  - مرحلهٔ ۵- $y\_in_2>0 \rightarrow y_2=1$
  - مرحلهٔ ۶- $y=(1,1,1,0)$
  - مرحلهٔ ۷- همگرایی شبکه را بررسی کنید.

تغییر چند فعال‌سازی در چرخهٔ به‌روز شدن ⇦ تکرار حداقل یک بار دیگر

عدم تغییر فعال‌سازی واحدها در تکرار دوم = همگرایی به بردار ذخیره شده

شبکهٔ هاپفیلد گسسته: تحلیل . . .

- هاپفیلد ثابت کرد
  - شبکهٔ گسستهٔ وی با در نظر گرفتن یک تابع انرژی (یا لیاپونف) برای سیستم، به نقطهٔ حدی پایدار (الگوی فعال‌سازی واحدها) همگرا خواهد شد.

- تابع انرژی (لیاپونف) . . .
  - یک تابع کران پایین و غیرصعودی از حالت سیستم
  - حالت سیستم = بردار فعال‌سازی‌های واحدها
  - تابع انرژی شبکهٔ هاپفیلد گسسته

$$E = -0.5\sum_{i\neq j}\sum_{j} y_i y_j w_{ij} - \sum_i x_i y_i + \sum_i \theta_i y_i$$

- مهم‌ترین ویژگی‌های این شبکه برای تضمین همگرایی
  - به‌روز کردن غیرهم‌زمان وزن‌ها
  - وجود وزن‌های صفر روی قطر اصلی ماتریس وزن

شبکهٔ هاپفیلد پیوسته . . .

- **شکل تغییر یافتهٔ شبکهٔ هاپفیلد گسسته با توابع خروجی پیوسته**
  - کاربرد در حل مسائل حافظهٔ انجمنی (همانند هاپفیلد گسسته) یا مسائل بهینه‌سازی با محدودیت (مثل مسئله فروشنده دوره‌گرد)
  - اتصالات بین واحدها، مشابه هاپفیلد گسسته، است و ماتریس وزن متقارن است
  - برای این شبکه، فعالیت درونی نرون = u_i و سیگنال خروجی = $v_i = g(u_i)$

تابع فعال‌سازی

- **تابع انرژی**

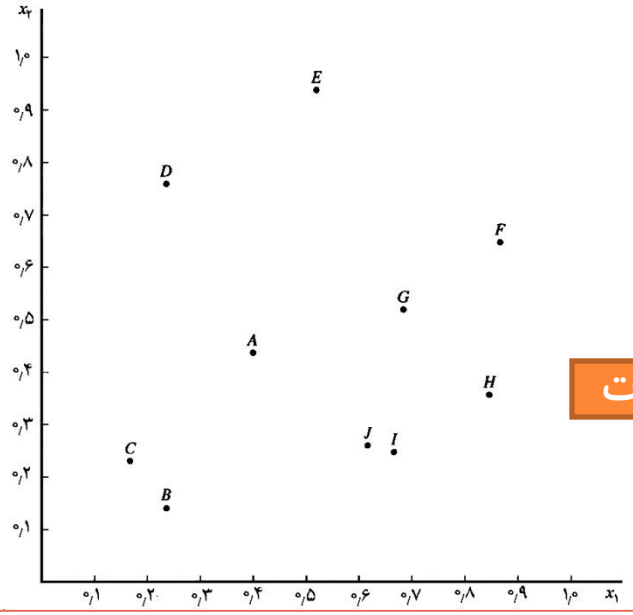
$$E = 0{,}5 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} v_i v_j + \sum_{i=1}^{n} \theta_i v_i$$

  - تا زمانی‌که $\frac{d}{dt} E \le 0$، شبکه به یک پیکره‌بندی ثابت (تابع انرژی کمینه است)، همگرا می‌شود.
  - شبکه با تابع انرژی فوق، زمانی همگرا می‌شود که فعالیت هر نرون طبق معادلهٔ دیفرانسیل زیر با زمان تغییر کند

$$\frac{d}{dt} u_i = -\frac{\partial E}{\partial v_i} = -\sum_{j=1}^{n} w_{ij} v_j - \theta_i$$

شبکهٔ هاپفیلد پیوسته: مثال . . .

- **مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) . . .**
  - یک مثال کلاسیک از بهینه‌سازی با محدودیت (Constrained Optimization)
  - یک فروشنده باید از n شهر مشخص بگذرد، از هرکدام از شهرها فقط یک‌بار عبور کند و در پایان سفر خود به شهر اول بازگردد.
  - هدف: سفری با خصوصیات فوق ولی با حداقل مسافت طی شده
  - یافتن راه‌حل مناسب با افزایش تعداد شهرها به سرعت مشکل می‌شود



مختصات

- در اینجا: یافتن راه‌حل با ۱۰ شهر

	x_1	x_2
A	0,4000	0,4439
B	0,2439	0,1463
C	0,1707	0,2293
D	0,2293	0,7610
E	0,5171	0,9414
F	0,8732	0,6536
G	0,6878	0,5219
H	0,8488	0,3609
I	0,6683	0,2536
J	0,6195	0,2634

شبکهٔ هاپفیلد پیوسته: مثال . . .

- مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) . . .
  - فاصلهٔ بین شهرها = ماتریس فاصله متقارن

	A	B	C	D	E	F	G	H	I	J
A	0,0000	0,3361	0,3141	0,3601	0,5111	0,5176	0,2982	0,4564	0,3289	0,2842
B	0,3361	0,0000	0,1107	0,6149	0,8407	0,8083	0,5815	0,6418	0,4378	0,3934
C	0,3141	0,1107	0,0000	0,5349	0,7919	0,8207	0,5941	0,6908	0,4982	0,4501
D	0,3601	0,6149	0,5349	0,0000	0,3397	0,6528	0,5171	0,7375	0,6710	0,6323
E	0,5111	0,8407	0,7919	0,3397	0,0000	0,4579	0,4529	0,6686	0,7042	0,6857
F	0,5176	0,8083	0,8207	0,6528	0,4579	0,0000	0,2274	0,2937	0,4494	0,4654
G	0,2982	0,5815	0,5941	0,5171	0,4529	0,2274	0,0000	0,2277	0,2690	0,2674
H	0,4564	0,6418	0,6908	0,7375	0,6686	0,2937	0,2277	0,0000	0,2100	0,2492
I	0,3289	0,4378	0,4982	0,6710	0,7042	0,4494	0,2690	0,2100	0,0000	0,0498
J	0,2842	0,3934	0,4501	0,6323	0,6857	0,4654	0,2674	0,2492	0,0498	0,0000

- راه‌حل مسئله با شبکه عصبی = یافتن کمینه تابع انرژی یا بیشینه تابع اجماع شبکه
- برتری شبکه عصبی به تکنیک‌های قدیمی = یافتن راه حل سریع (تقریباً بهینه) برای مسائل بزرگ

H. Veisi (h.veisi@ut.ac.ir)

شبکهٔ هاپفیلد پیوسته: مثال . . .

- **مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) ساختار شبکه . . .**
  - **برای n شهر، n*n نرون نیاز است**

	Position									
City	1	2	3	4	5	6	7	8	9	10
A	$U_{A,1}$	$U_{A,2}$	$U_{A,3}$	$U_{A,4}$	$U_{A,5}$	$U_{A,6}$	$U_{A,7}$	$U_{A,8}$	$U_{A,9}$	$U_{A,10}$
B	$U_{B,1}$	$U_{B,2}$	$U_{B,3}$	$U_{B,4}$	$U_{B,5}$	$U_{B,6}$	$U_{B,7}$	$U_{B,8}$	$U_{B,9}$	$U_{B,10}$
C	$U_{C,1}$	$U_{C,2}$	$U_{C,3}$	$U_{C,4}$	$U_{C,5}$	$U_{C,6}$	$U_{C,7}$	$U_{C,8}$	$U_{C,9}$	$U_{C,10}$
D	$U_{D,1}$	$U_{D,2}$	$U_{D,3}$	$U_{D,4}$	$U_{D,5}$	$U_{D,6}$	$U_{D,7}$	$U_{D,8}$	$U_{D,9}$	$U_{D,10}$
E	$U_{E,1}$	$U_{E,2}$	$U_{E,3}$	$U_{E,4}$	$U_{E,5}$	$U_{E,6}$	$U_{E,7}$	$U_{E,8}$	$U_{E,9}$	$U_{E,10}$
F	$U_{F,1}$	$U_{F,2}$	$U_{F,3}$	$U_{F,4}$	$U_{F,5}$	$U_{F,6}$	$U_{F,7}$	$U_{F,8}$	$U_{F,9}$	$U_{F,10}$
G	$U_{G,1}$	$U_{G,2}$	$U_{G,3}$	$U_{G,4}$	$U_{G,5}$	$U_{G,6}$	$U_{G,7}$	$U_{G,8}$	$U_{G,9}$	$U_{G,10}$
H	$U_{H,1}$	$U_{H,2}$	$U_{H,3}$	$U_{H,4}$	$U_{H,5}$	$U_{H,6}$	$U_{H,7}$	$U_{H,8}$	$U_{H,9}$	$U_{H,10}$
I	$U_{I,1}$	$U_{I,2}$	$U_{I,3}$	$U_{I,4}$	$U_{I,5}$	$U_{I,6}$	$U_{I,7}$	$U_{I,8}$	$U_{I,9}$	$U_{I,10}$
J	$U_{J,1}$	$U_{J,2}$	$U_{J,3}$	$U_{J,4}$	$U_{J,5}$	$U_{J,6}$	$U_{J,7}$	$U_{J,8}$	$U_{J,9}$	$U_{J,10}$

- **سفر صحیح =وجود دقیقاً یک واحد «فعال» در هر ردیف و در هر ستون**
  - وجود دو واحد «فعال» در یک ردیف = از شهر متناظر آن ردیف دو بار عبور شده است
  - وجود دو واحد «فعال» در یک ستون = فروشنده هم‌زمان در دو شهر حضور داشته است

شبکهٔ هاپفیلد پیوسته: مثال . . .

- **مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) . . .**
  - وزن‌های شبکه طوری تعیین می‌شوند که واحدهای قرار گرفته در یک ردیف (یا یک ستون) به طور هم‌زمان «فعال» نباشند
  - بین واحدهای ستون‌های مجاور و بین واحدهای ستون‌های اول و آخر، اتصالاتی مطابق با فاصلهٔ بین شهرها وجود دارد

  - تابع انرژی هاپفیلد-تانک برای مسئله فروشنده دوره‌گرد

$$E = \frac{A}{2}\sum_{x}\sum_{i}\sum_{j\neq i} v_{x,i}v_{x,j} + \frac{B}{2}\sum_{i}\sum_{x}\sum_{y\neq x} v_{x,i}v_{y,j} + \frac{C}{2}\left[N - \sum_{x}\sum_{i} v_{x,i}\right]^2 + \frac{D}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{x,y}v_{x,i}(v_{y,i+1}+v_{y,i-1})$$

شهر — موقعیت

  - تابع سیگموید برای تولید سیگنال خروجی (برد ۰ و ۱)

$$v_i = g(u_i) = 0{,}5\,[1+\tanh(\alpha u_i)]$$

شبکهٔ هاپفیلد پیوسته: مثال . . .

- **مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) . . .**
  - معادله دیفرانسیل برای فعالیت واحد $U_{X,I}$

$$\frac{d}{dt}u_{X,I} = -\frac{u_{X,I}}{\tau} - A\sum_{j\neq I} v_{X,j} - B\sum_{y\neq X} v_{y,I} + C\left[N - \sum_{x}\sum_{i} v_{x,i}\right] - D\sum_{y\neq X} d_{X,y}(v_{y,I+1} + v_{y,I-1})$$

وزن‌های مربوط به اتصالات بین ردیف‌ها

اتصالات بین ستون‌ها

اتصالات مربوط به فاصله بین شهرها

  - وزن‌های بین واحدهای U_{xi} و U_{yj} (ثابت)

$$w(x,i;y,i) = -A\delta_{xy}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{xy}) - C - Dd_{xy}(\delta_{ij+1} + \delta_{ij-1}),$$

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$ دلتای دیراک

N = مقداری بزرگ‌تر از تعداد شهرها، n

  - ورودی خارجی برای هر واحد $I_{xi} = +CN$

شبکهٔ هاپفیلد پیوسته: مثال . . .

- **مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) . . .**
  - مرحلهٔ ۰- مقدار اولیهٔ فعال‌سازی تمام واحدها را تعیین کنید.
    مقدار اولیهٔ Δt را عدد کوچکی قرار دهید.
  - مرحلهٔ ۱-تا زمانی‌که شرایط توقف غلط است، مراحل ۲ تا ۶ را انجام دهید.
  - مرحلهٔ ۲- مراحل ۳ تا ۵ را n^2 بار (n تعداد شهرها است) انجام دهید.
  - مرحلهٔ ۳- یک واحد را به طور تصادفی انتخاب کنید.
  - مرحلهٔ ۴- فعالیت واحد انتخاب شده را تغییر دهید:

$$u_{x,i}(new) = u_{x,i}(old) + \Delta t\left[-u_{x,i}(old) - A\sum_{j\neq i} v_{x,j} - B\sum_{y\neq x} v_{y,i} + C\left(N - \sum_x\sum_j v_{x,j}\right) - D\sum_{y\neq x} d_{x,y}(v_{y,i+1} + v_{y,i-1})\right]$$

  - مرحلهٔ ۵- تابع خروجی را اعمال کنید: $v_{x,i} = 0/5\,[1+\tanh(\alpha u_{x,i})]$
  - مرحلهٔ ۶- شرایط توقف را بررسی کنید.

شبکهٔ هاپفیلد پیوسته: مثال . . .

- **مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man) . . .**
  - مقادیر پارامترهای مورد استفاده توسط هاپفیلد-تانک

$$A = B = D = 500,\ C = 200,\ N = 15,\ \alpha = 50$$

  - مقدار بزرگ α = تابع سیگموید با شیب زیاد = تقریبی از تابع پله
  - سطح فعالیت‌های اولیه واحدها طوری انتخاب می‌شوند که $\sum_x \sum_i u_{x,i} = 10$ (معادل کل فعال‌سازی مطلوب برای یک سفر صحیح)

شبکهٔ هاپفیلد پیوسته: مثال

- مثال: مسئلهٔ فروشنده دوره‌گرد (Travel Sales Man)
  - نتیجه: تقریباً در نصف تست‌ها، یکی از دو کوتاه‌ترین مسیر را تولید کرده‌اند
  - بهترین سفر به دست آمده A D E F G H J I B C
    - طول = ۲.۷۱