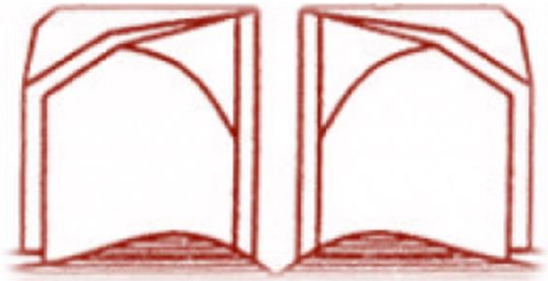


مبانی رایانش نرم

شبکه‌های عصبی: پرسپترون، آدالاین

هادی ویسی

h.veisi@ut.ac.ir

دانشگاه تهران - دانشکده علوم و فنون نوین

فهرست

- شبکه پرسپترون
  - ساختار
  - الگوریتم یادگیری
  - کاربردها و مثال
  - همگرایی قانون یادگیری
- شبکه آدالاین
  - ساختار
  - الگوریتم یادگیری
  - کاربردها و مثال
  - قانون دلتا

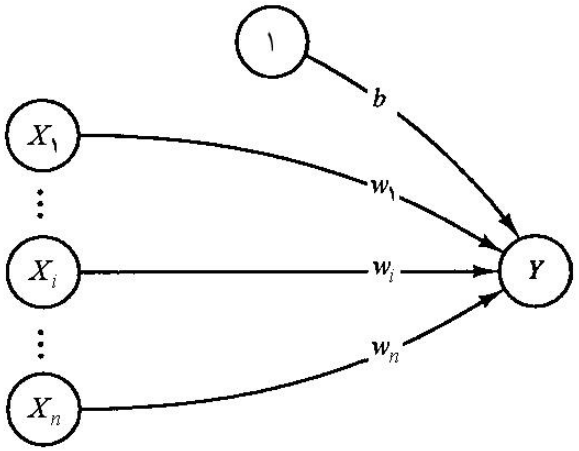
شبکه پرسپترون . . .

- 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.

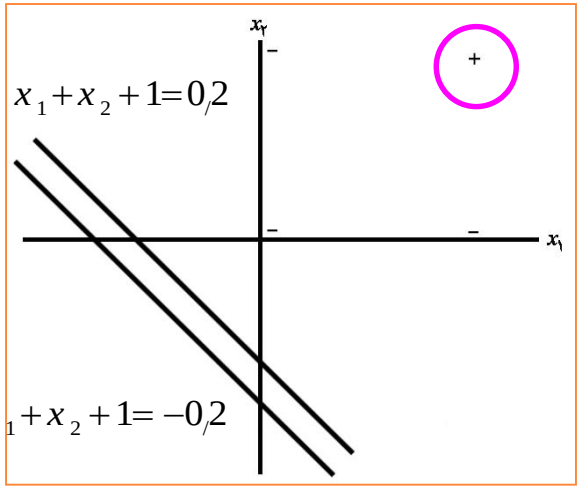
خطا = برابر نبودن پاسخ شبکه و مقدار هدف

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

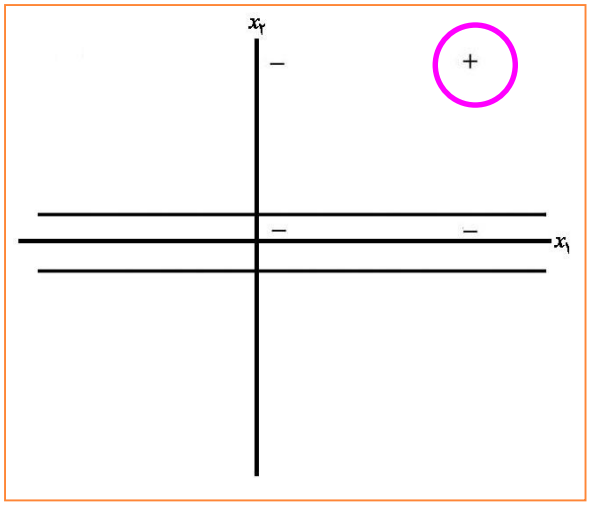
INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
$(x_1$	x_2	$1)$	y_in	y	t	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	w_2	$b)$
									$(0$	0	$0)$
$(1$	1	$1)$	0	0	1	$(1$	1	$1)$	$(1$	1	$1)$



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	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	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	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	w_2	$b)$
									$(0$	0	$-1)$
$(0$	0	$1)$	-1	-1	-1	$(0$	0	$0)$	$(0$	0	$-1)$

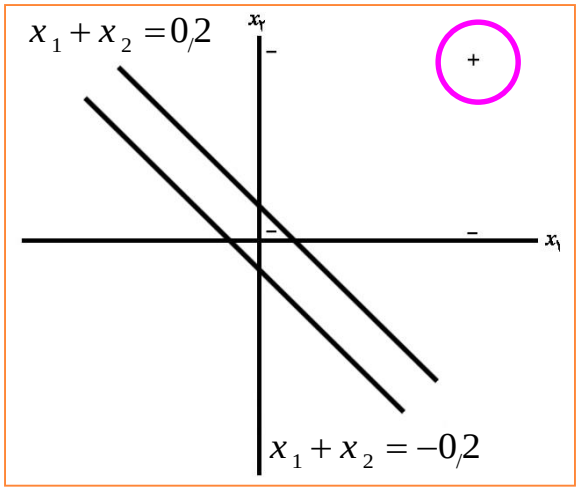
کامل شدن اولین دور آموزش (Epoch)

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

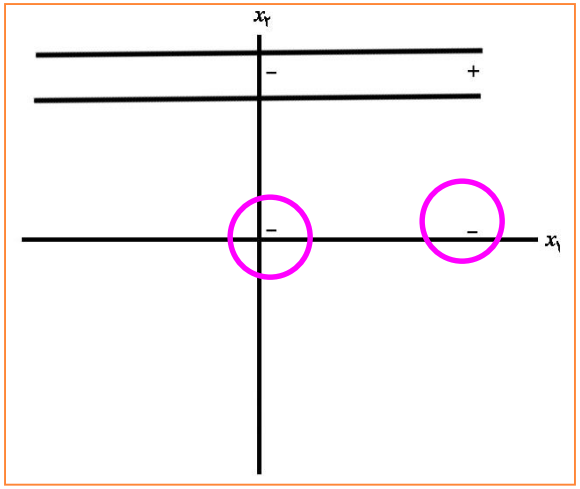
INPUT			NET	OUT	TARGET	WEIGHT CHANGES			WEIGHTS		
$(x_1$	x_2	$1)$	y_in	y	t	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	w_2	$b)$
									$(0$	0	$-1)$
$(1$	1	$1)$	-1	-1	1	$(1$	1	$1)$	$(1$	1	$0)$



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	$(\Delta w_1$	Δw_2	$\Delta b)$	$(w_1$	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 . . .

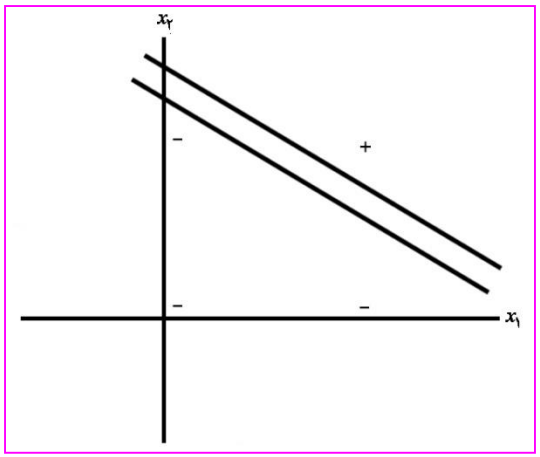
- تابع AND با ورودی‌های دودویی و هدف‌های دوقطبی . . .
  - پنجمین، ششمین، .... دور آموزش

(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)

  - نهمین دور آموزش

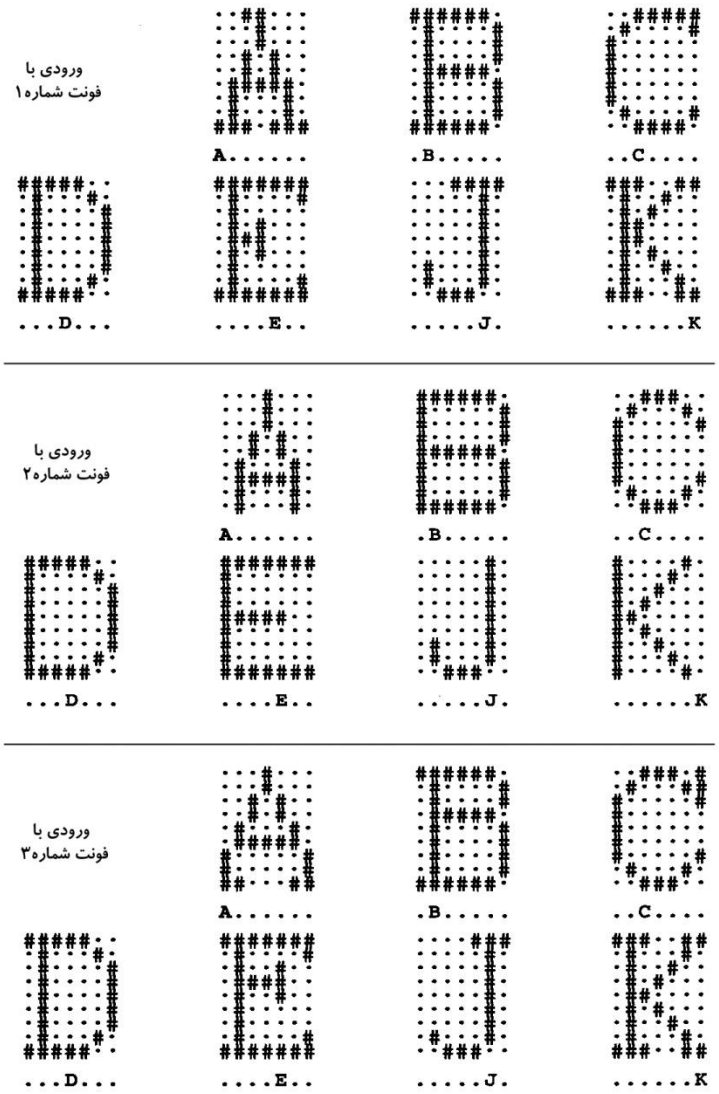
(1	1	1)	1	1	1	(0	0	0)	(2	3	–4)
(1	0	1)	–2	–1	–1	(0	0	0)	(2	3	–4)
(0	1	1)	–1	–1	–1	(0	0	0)	(2	3	–4)
(0	0	1)	–4	–1	–1	(0	0	0)	(2	3	–4)

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



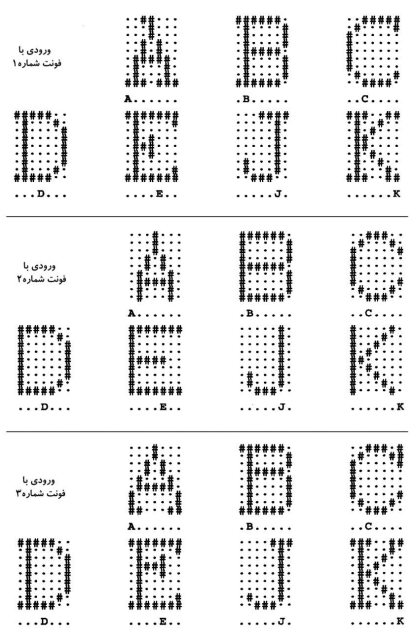
$$\begin{cases} 2x_1 + 3x_2 - 4 > 0{,}2 \Rightarrow x_2 = -\frac{2}{3}x_1 + \frac{7}{5} \\ 2x_1 + 3x_2 - 4 < -0{,}2 \Rightarrow x_2 = -\frac{2}{3}x_1 + \frac{19}{15} \end{cases}$$

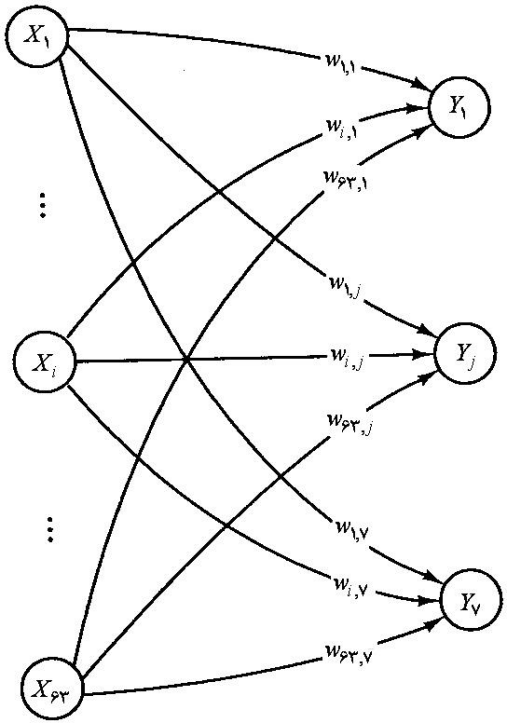
شبکه پرسپترون: مثال . . .



- 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

$$(\mathrm{A}.....) \Rightarrow (1,-1,-1,-1,-1,-1,-1)$$

    - بردار خروجی برای نمونه‌های B

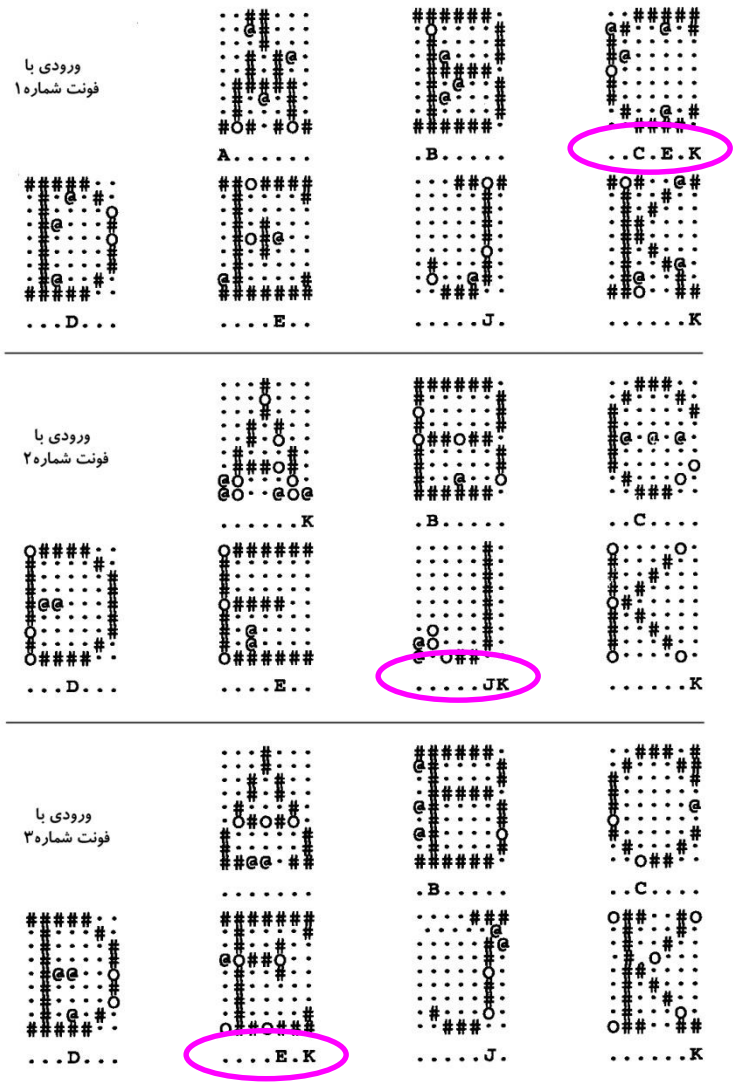
$$(.\,\mathrm{B}....) \Rightarrow (-1,1,-1,-1,-1,-1,-1)$$

- 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

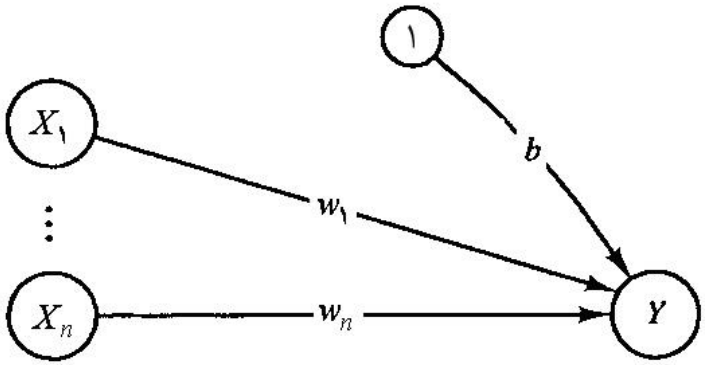
- مقداردهی نرخ یادگیری
  - مقدار ثابت غیرمنفی
  - مقدار $1/\|x\|$ ؛ تا تغییر وزن یک بردار واحد باشد
  - مقدار $(x.w)/\|x\|^2$
- مقادیر اولیه وزن‌ها
  - مقدار ثابت صفر
  - یک الگوی آموزش اختیاری
  - مقادیر تصادفی کوچکی

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 بزرگ فرض کرده (مثلاً ۰.۹) و به مرور مقدار آن را کوچک 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

x_1	x_2	t
1	1	1
1	0	−1
0	1	−1
0	0	−1

- شبکه بعد از آموزش

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

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

- 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

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

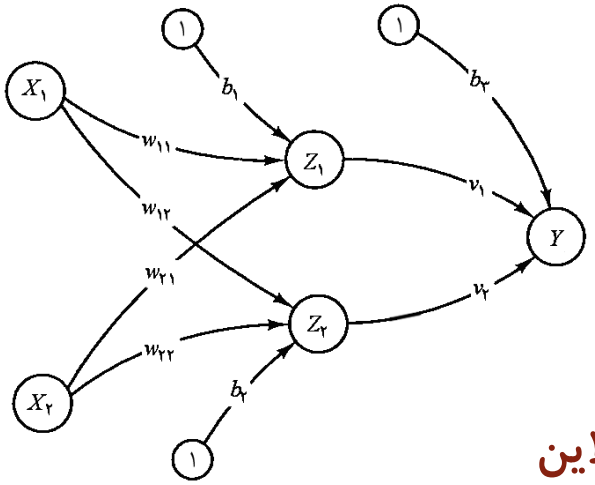
$$\Delta w_I = \alpha(t - y\_in)x_I$$

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