

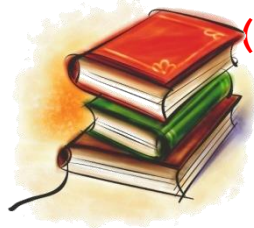
# مبانی رایانش نرم

شبکه‌های عصبی: مقدمه. شبکه مک‌کلاچ-پیتز و شبکه هب

هادی ویسی

[h.veisi@ut.ac.ir](mailto:h.veisi@ut.ac.ir)

دانشگاه تهران - دانشکده علوم و فنون نوین



- شبکه عصبی چیست؟
  - شبکه‌های عصبی طبیعی
- کاربردهای شبکه‌های عصبی مصنوعی
- تاریخچه شبکه‌های عصبی مصنوعی
- تعاریف
- شبکه مک کلاچ - پیتز
  - ساختار، الگوریتم، کاربرد، مثال
- شبکه هب
  - الگوریتم، کاربردها و مثال
- جداسازی خطی

## شبکه عصبی؟ ...

○ کارهایی که مغز انسان انجام می‌دهد

- یادگیری (تشخیص چهره)

- ذخیره‌سازی اطلاعات

- تصمیم‌گیری

- پیش‌بینی

- محاسبه

- ...



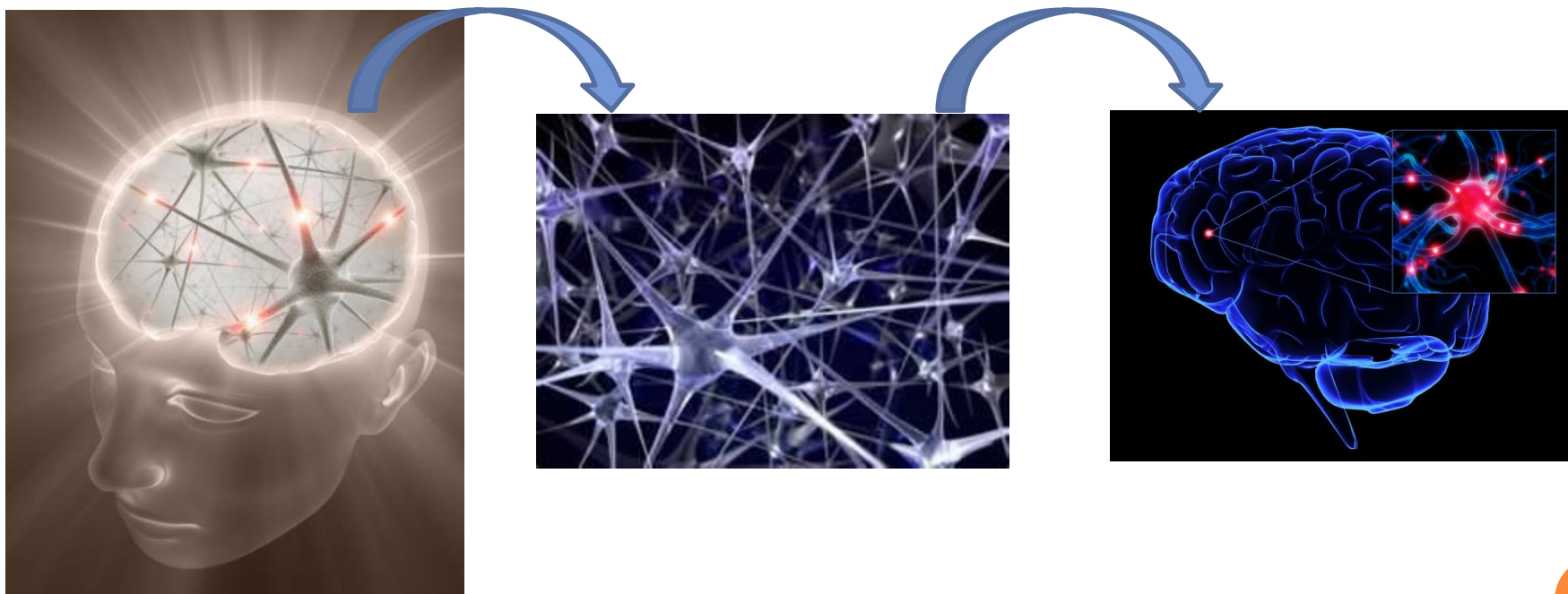
## شبکه عصبی؟

○ مغز = شبکه‌ای بسیار بزرگ از عصب‌ها (نرون‌ها)

• ۱۰۰.۰۰۰.۰۰۰.۰۰۰ نرون

• ۱۰.۰۰۰ اتصال برای هر نرون

○ شبکه عصبی مصنوعی = شبیه‌سازی شبکه عصبی طبیعی



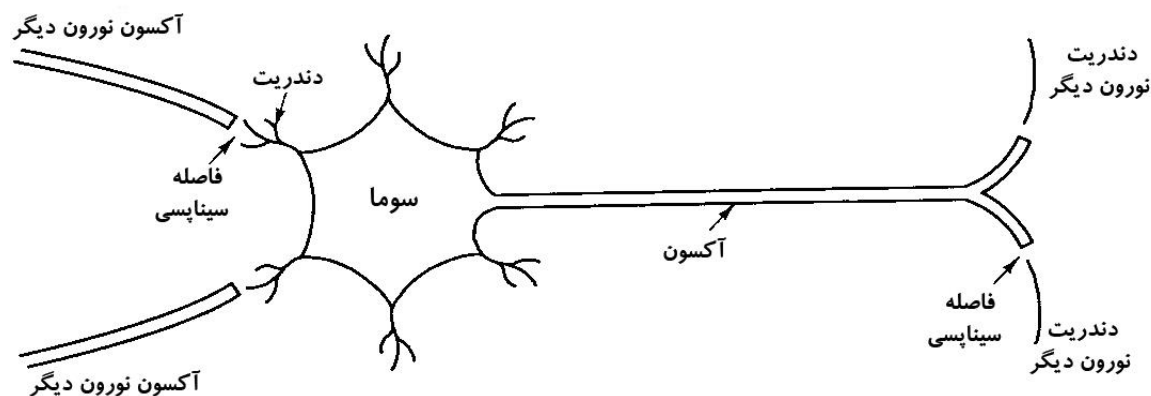
## شبکه عصبی طبیعی ...

### ○ عنصر پردازشگر تشکیل دهنده یک شبکه عصبی مصنوعی

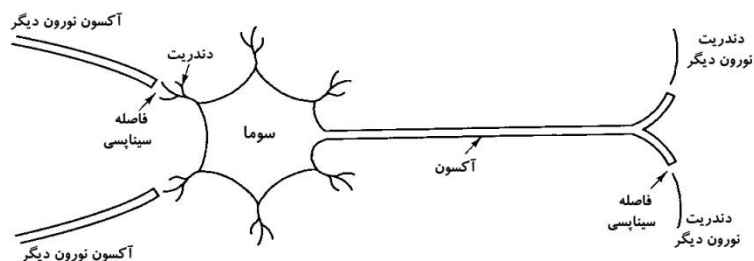
• نرون (Neuron) = عصب طبیعی (سلول مغزی)

### ○ سه جزء تشکیل دهنده یک نرون طبیعی

- دندریت‌ها (Dendrite): دریافت سیگنال از سایر نرون‌ها
- سوما (Soma) = بدنه سلول: سیگنال‌های ورودی به سلول را جمع می‌بندد
- آکسون (Axon): ارسال سیگنال به نرون(های) دیگر



## شبکه عصبی طبیعی ...



### عملکرد نرون طبیعی

- دریافت سیگنال از سایر نرون‌ها توسط دندریت‌ها
- عبور سیگنال‌ها با یک فرآیند شیمیایی از فاصله سیناپسی (Synaptic Gap)
- عمل شیمیایی انتقال دهنده، سیگنال ورودی را تغییر می‌دهند (تضعیف/تقویت سیگنال)
- سوما سیگنال‌های ورودی به سلول را جمع می‌بندد
- زمانی که یک سلول به اندازه کافی ورودی دریافت نماید، برانگیخته می‌شود و سیگنالی را از آکسون خود به سلول‌های دیگر می‌فرستد.
- انتقال سیگنال از یک نرون خاص نتیجه غلظت‌های مختلف یون‌ها در اطراف پوشش آکسون نرون («ماده سفید» مغز) می‌باشد.
- یون‌ها = پتاسیم، سدیم و کلرید
- سیگنال‌ها به صورت ضربه‌های الکتریکی هستند

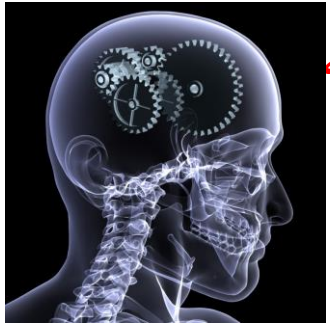




## شبکه عصبی طبیعی ...

### ○ خلاصه ویژگی‌ها و خصوصیات نرون‌های طبیعی

- جزء پردازشگر (نرون) سیگنال‌های فراوانی را دریافت می‌کند.
- سیگنال‌های ورودی ممکن است با یک وزن در سیناپس سلول دریافت کننده، تغییر کند.
- جزء پردازشگر ورودی‌های وزن‌دار را جمع می‌بندد.
- نرون در شرایط مناسب (ورودی کافی)، یک سیگنال را به عنوان خروجی انتقال می‌دهد.
- خروجی یک نرون ممکن است به بسیاری از نرون‌های دیگر (شاخه‌های آکسون) برود.
- پردازش اطلاعات به صورت محلی صورت می‌گیرد.
- مفهوم حافظه در اجزای مختلف سلول توزیع می‌شود:
  - حافظه بلند مدت در سیناپس‌ها یا وزن‌های نرون قرار می‌گیرد.
  - حافظه کوتاه مدت با سیگنال‌های فرستاده شده توسط نرون‌ها مطابقت دارد.
- توانایی سیناپس می‌تواند با آزمایش و کسب تجربه تغییر کند.
- انتقال دهنده‌های عصبی برای سیناپس‌ها می‌توانند تحریک‌کننده (Excitatory) یا بازدارنده (Inhibitory) باشند.



## شبکه عصبی مصنوعی ...

### ○ شبکه عصبی مصنوعی (Artificial Neural Network)

- یک سیستم پردازش اطلاعات با ویژگی‌های مشترکی با شبکه‌های عصبی طبیعی
- تعمیم یافته مدل‌های ریاضی تشخیص انسان بر اساس زیست‌شناسی عصبی

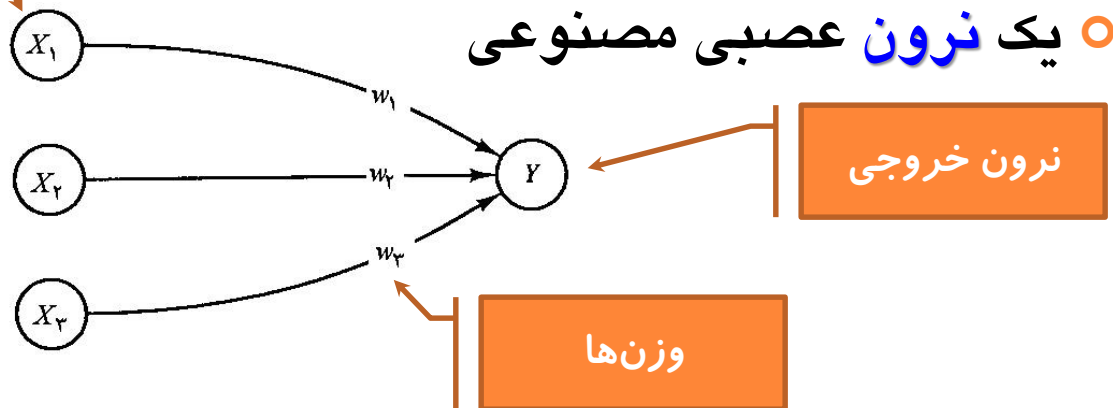
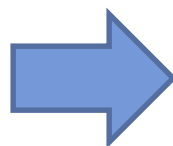
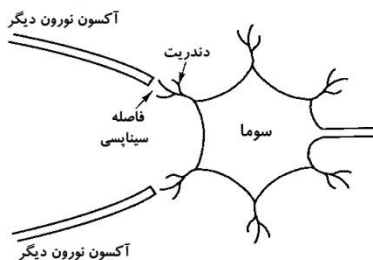
### ○ فرضیات پایه شبکه عصبی مصنوعی

- پردازش اطلاعات در اجزای ساده‌ای با تعداد فراوان، به نام نرون‌ها صورت می‌گیرد.
  - سیگنال‌ها در بین نرون‌های شبکه از طریق پیوندها یا اتصالات (Connections) آنها منتقل می‌شوند.
  - هر پیوند، وزن (Weight) مربوط به خود را دارد که در شبکه‌های عصبی رایج در سیگنال‌های انتقال یافته از آن پیوند ضرب می‌شود.
  - هر نرون یک تابع فعال‌سازی (Activation Function) را بر روی ورودی‌های خود اعمال می‌کند تا سیگنال خروجی خود را تولید نماید.
- تابع معمولاً غیرخطی است



## شبکه عصبی مصنوعی ...

نرون‌های ورودی



- فعال‌سازی‌ها یا سیگنال‌های خروجی نرون‌های ورودی به ترتیب  $x_1$ ،  $x_2$  و  $x_3$  هستند
- ورودی شبکه به نرون  $Y$ ، حاصل جمع وزن دار سیگنال‌های ورودی و وزن‌هاست:

$$y_{in} = w_1 x_1 + w_2 x_2 + w_3 x_3 = \sum_i w_i x_i$$

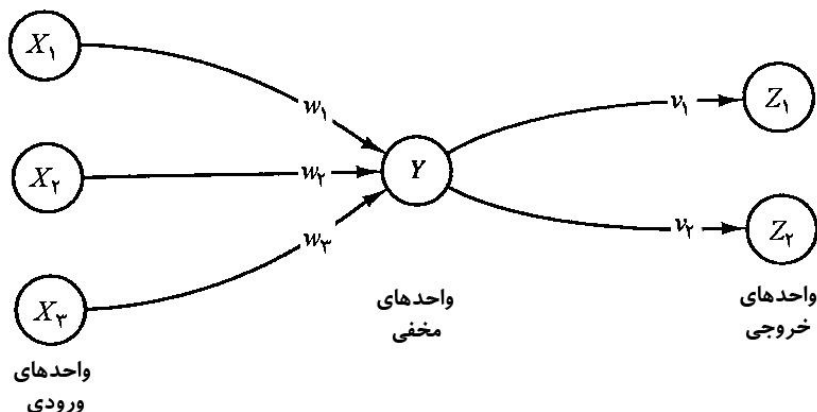
- فعال‌سازی نرون  $Y$  با اعمال تابع فعال‌سازی  $f$  روی ورودی آن به دست می‌آید
- تابع سیگموید (Sigmoid)

$$y = f(y_{in})$$

$$f(x) = \frac{1}{1 + \exp(-x)}$$

## شبکه عصبی مصنوعی ...

### یک شبکه عصبی مصنوعی



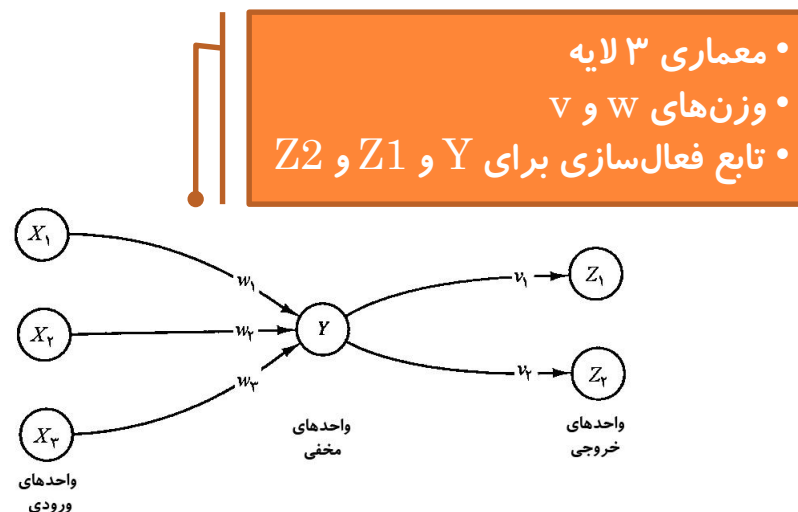
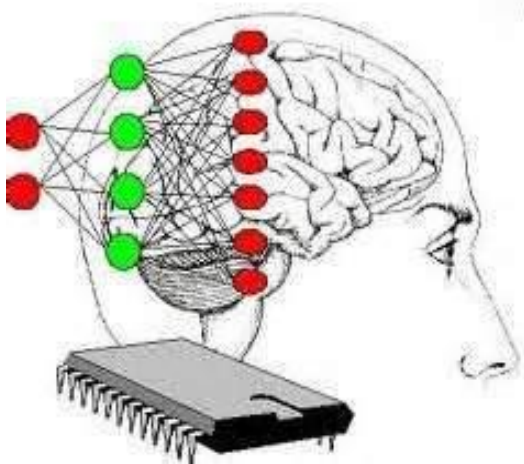
- سه لایه: ورودی، مخفی و خروجی
- دو دسته وزن:  $w$  ها و  $v$  ها

- در یک شبکه یک نرون می‌تواند ورودی‌های مختلفی را از چند نرون دریافت کند

## شبکه عصبی مصنوعی

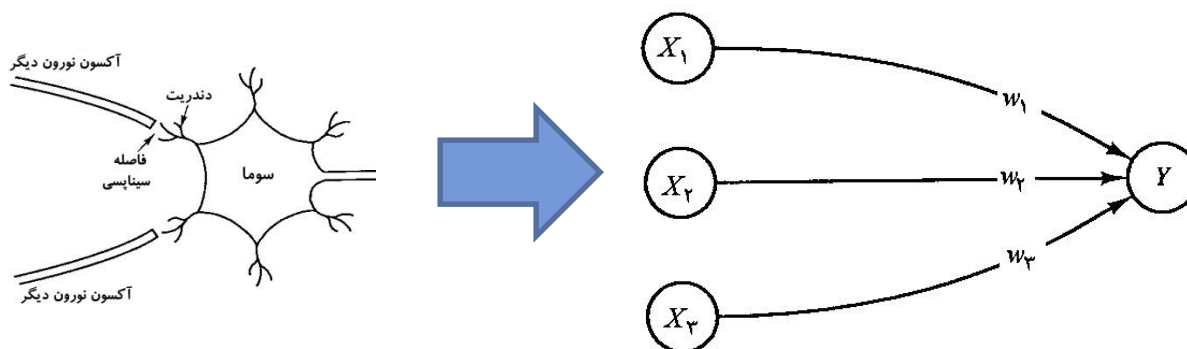
### ویژگی‌های مشخص کننده یک شبکه عصبی مصنوعی

- ساختار یا معماری شبکه (Architecture): الگوی پیوندهای بین نرون‌های مختلف
- الگوریتم آموزش یا یادگیری (Training or Learning Algorithm): روش تعیین وزن‌های روی پیوندهای شبکه
- تابع فعال‌سازی شبکه (Activation Function) که هر نرون روی ورودی‌های خود اعمال می‌کند



## شباهت شبکه‌های عصبی طبیعی و مصنوعی ...

شبکه عصبی مصنوعی	شبکه عصبی طبیعی
اتصالات بین نرون‌ها	دندريت
وزن‌های شبکه	تغيير سيگنال ورودی هنگام عبور از فاصله سيناپسی
جمع وزن‌دار سيگنال‌های ورودی و وزن در نرون	جمع بستن سيگنال‌های ورودی در سوما
تابع فعال‌سازی	برانگیخته شدن سلول و ارسال سيگنال از آکسون



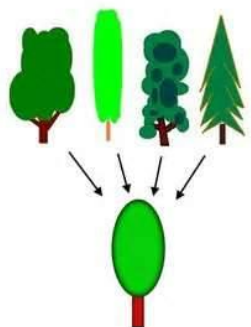
## شباهت شبکه‌های عصبی طبیعی و مصنوعی

### ویژگی‌های مهم مشترک

- تحمل‌پذیری در برابر خطا (Fault Tolerance)  
○ از بین رفتن تعداد زیادی از نرون‌های طبیعی در طول زمان اما یادگیری ادامه می‌یابد

### تعمیم‌پذیری (Generalization) و مقاوم بودن در برابر نویز

- تشخیص سیگنال‌های ورودی که با سیگنال قبلاً مشاهده شده تا حدودی متفاوت است  
○ تشخیص چهره، تشخیص دستخط و ...



- پردازش‌های موازی با تعداد زیادی از واحدهای پردازشگر

## کاربردهای شبکه‌های عصبی مصنوعی ...

### ○ پزشکی

- ذخیره‌سازی حجم زیادی از اطلاعات پزشکی

- ورودی مجموعه‌ای از علائم یک بیماری خاص
- خروجی: پیدا کردن «بهترین» تشخیص و نحوه درمان آن با استفاده از الگوی ذخیره شده متناسب با علائم آن بیماری
- شبکه عصبی حافظه خودانجمنی



### ○ تولید گفتار (خواندن متن)

- تبدیل متن به گفتار برای خواندن متن

- سیستم NETtalk
- ورودی: حروف متن (هر حرف به همراه سه حرف قبل و بعد از آن)
- خروجی: صدای مربوط به آن حرف
- شبکه عصبی چندلایه



## کاربردهای شبکه‌های عصبی مصنوعی ...

### ○ پردازش سیگنال

- حذف نویز در سیگنال صدا (مکالمه تلفن)
- حذف نویز به صورت افقی - Adaptive Noise Cancellation (ANC)
- شبکه عصبی آدالین
- حذف انعکاس صدا (اکو)

### ○ کنترل

- کنترل دمای اتاق
- مسیر حرکت ضد موشک
- دنده عقب رفتن کامیون
- شبکه پس انتشار بازگشتی

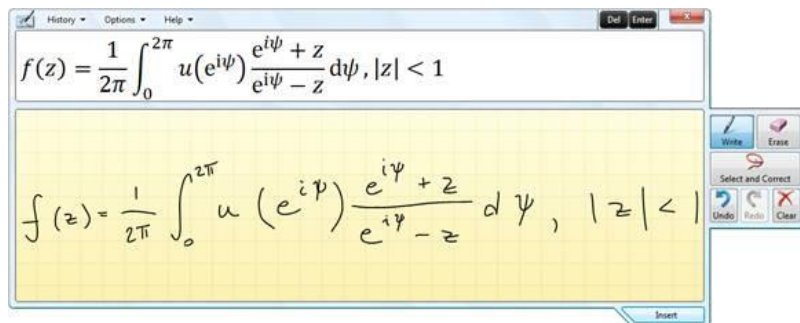


## کاربردهای شبکه‌های عصبی مصنوعی ...

### ○ بازشناسی الگو (تشخیص الگو) ...

• بازشناسی خودکار دست خط

○ شبکه‌های پسانتشار چندلایه



• بازشناسی نویسه‌های نوری (Optical Character Recognition: OCR)

○ شبکه Neocognitron



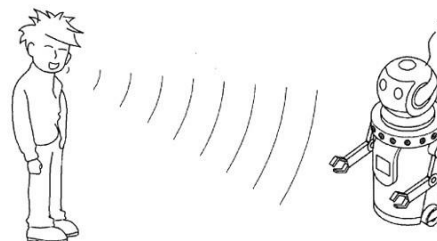
## کاربردهای شبکه‌های عصبی مصنوعی ...

### ○ بازشناسی الگو (تشخیص الگو)

#### • بازشناسی خودکار گفتار (Automatic Speech Recognition: ASR)

○ شبکه‌های چندلایه با اتصالات بازگشتی

○ نگاشت خودسازمانده کوهونن



#### • بازشناسی چهره (Face Recognition)

## کاربردهای شبکه‌های عصبی مصنوعی ...

### تجارت

#### • ارزیابی میزان خطرپذیری وام‌دهی

- ورودی: سال‌های اشتغال متقاضی، تعداد افراد تحت تکفل، درآمد فعلی و ویژگی‌های خود وام (مثل مبلغ، نرخ سود و ...)
- خروجی: پاسخ «قبول» یا «رد» برای دادن وام



### پیش‌بینی

- مصرف برق کشور در سه ماه آینده
- وضعیت آب و هوا
- سود سهام



## تاریخچه شبکه‌های عصبی مصنوعی ...

### ○ دهه ۴۰ - اولین شبکه‌های عصبی مصنوعی

- ۱۹۴۳ - معرفی نرون مک کلاچ-پیتز (اولین شبکه عصبی مصنوعی)

○ توسط وارن مک کلاچ و والتر پیتز در ۱۹۴۳ و توسعه در ۱۹۴۷

- ۱۹۴۹ - شبکه هب

○ توسط دونالد هب، یکی از روانشناسان دانشگاه McGill

○ ایده: اگر دو نرون به طور هم‌زمان فعال شوند، استحکام اتصال بین آنها باید افزایش یابد

○ اولین قانون یادگیری

### ○ دهه ۵۰ - پرسپترون

- معرفی توسط فرانک روزنبلات در سال ۱۹۵۸ و بهبود در ۱۳۵۹ و ۱۳۶۲

- شبکه لایه با الهام از شبکیه چشم

- قانون یادگیری قوی‌تر از قانون هب، مبتنی بر روشی تکرار شونده برای تنظیم وزن



## تاریخچه شبکه‌های عصبی مصنوعی ...

### ○ دهه ۶۰ - گسترش پرسپترون + آدالاین

#### • ۱۹۶۰ - شبکه آدالاین

- آدالاین (ADALINE) = نرون خطی افقی (ADaptive LInear NEuron) یا سیستم خطی افقی (ADaptive LINEar System)
- توسط برنارد ویدرو و دانشجوی وی مارسیان (تد) هاف
- ارائه یک قانون یادگیری با نام قانون ویدرو-هاف (Widrow-Hoff Rule) یا میانگین مربعات کمینه (LMS) و یا قانون دلتا (Delta Rule)
- شباهت زیاد قانون یادگیری دلتا (مهندسی) با قانون پرسپترون (روانشناسی)
- تفاوت: در پرسپترون برای هر واحدی که پاسخ نادرست دارد، وزن‌های اتصال آن واحد تنظیم می‌شود، اما قانون دلتا وزن‌ها را طوری تنظیم می‌کند تا اختلاف بین خروجی شبکه و خروجی مطلوب کمینه کند
- مادالاین: شکل توسعه یافته و چندلایه آدالاین
- قانون دلتا منجر به افزایش قابلیت تعمیم می‌شود
- قانون دلتا مبنای قانون پس‌انتشار (Backpropagation) برای یادگیری شبکه‌های چندلایه است

#### • ۱۹۶۹ - تشریح کامل پرسپترون توسط مینسکی و پاپرت





## تاریخچه شبکه‌های عصبی مصنوعی ...

### ○ دهه ۷۰ - سال‌های خاموش

- عدم موفقیت پرسپترون‌های یک لایه در حل مسائل ساده‌ای (مانند تابع XOR)
- عدم وجود روشی کلی برای آموزش شبکه‌های چندلایه
- ۱۹۷۲ - اولین کار کوهونن از دانشگاه هلسینکی، روی شبکه‌های عصبی حافظه پیوندی
- ۱۹۷۷ - تحقیقات آندرسن از دانشگاه براون در زمینه شبکه‌هایی عصبی حافظه انجمنی و انتشار نظریاتش با نام «حالت مغز در یک جعبه» (Brain-State-in-a-Box)



## تاریخچه شبکه‌های عصبی مصنوعی ...

### ○ دهه ۸۰ - شکوفایی شبکه‌های عصبی ...

- الگوریتم پسانتشار خطا برای آموزش شبکه‌های چندلایه

- توسط پارکر در سال ۱۹۸۵ و لوکان در سال ۱۹۸۶

- شبکه‌های هاپفیلد

- توسط هاپفیلد برنده جایزه نوبل در رشته فیزیک و عضو مؤسسه فن آوری کالیفرنیا

- به همراه دیوید تانک، محقق AT&T

- شبکه عصبی با وزن‌های ثبات و فعال‌سازی افقی (جزو شبکه‌های حافظه انجمنی)

- حل مسائل ارضای محدودیت مانند «مسئله فروشنده دوره گرد»

- نگاشت‌های خودسازمانده کوهونن (SOM)

- توسط کوهونن از دانشگاه هلسینکی

- استفاده در بازشناسی گفتار کلمات فنلاندی و ژاپنی، حل «مسئله فروشنده دوره گرد» و آهنگ‌سازی



## تاریخچه شبکه‌های عصبی مصنوعی ...

### ○ دهه ۸۰ - شکوفایی شبکه‌های عصبی ...

- شبکه‌های نظریهٔ نوسان وفقی (ART)

- توسط کارپنز و با همکاری گراس برگ
- نظریه‌ای برای شبکه‌های عصبی خودسازمانده

- شبکه Neocognitron

- توسط فوکوشیما و همکارانش در آزمایشگاه‌های NHK در توکیو
- شبکهٔ عصبی خاص منظوره برای بازشناسی نویسه‌ها
- بهبود یافته شبکهٔ خودسازمانده قدیمی‌تر با نام Cognitron (۱۹۷۵)

- ماشین بولتزمن

- تغییر وزن‌ها یا فعال‌سازی براساس تابع تراکم احتمال
- استفاده از ایده‌های کلاسیک شبیه‌سازی سردشدن تدریجی (Simulated Annealing) و تئوری تصمیم‌گیری بیز (Bayesian Decision Theory)

## تاریخچه شبکه‌های عصبی مصنوعی ...

### ○ دهه ۸۰ - شکوفایی شبکه‌های عصبی ...

- مطالعات ریاضیاتی و زیست‌شناختی

- گراس برگ (مدیر مرکز سیستم‌های وفقی در دانشگاه بوستون)

- پیاده‌سازی سخت‌افزاری

- افزایش قابلیت‌های محاسباتی کامپیوترها و ساخت VLSI برای شبکه‌های عصبی

- ایجاد شرکت‌های مبتنی بر شبکه عصبی





## تاریخچه شبکه‌های عصبی مصنوعی

### ○ دهه ۹۰ - دهه کاربرد

- به کارگیری شبکه‌های عصبی در کاربردهای مختلف
- توسعه شبکه توابع پایه شعاعی (RBF)
- ماشین بردار پشتیبان (SVM)

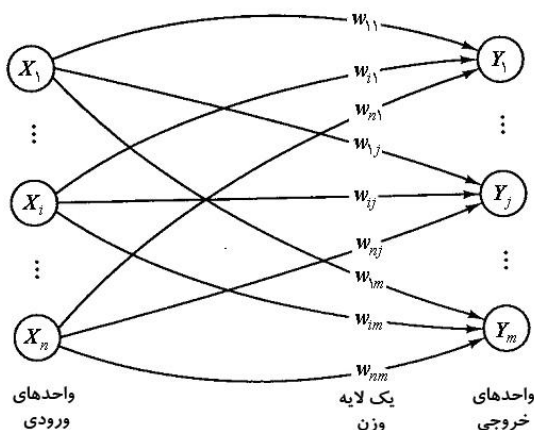
### ○ ۲۰۰۰ به بعد

- یادگیری عمیق (Deep Learning)

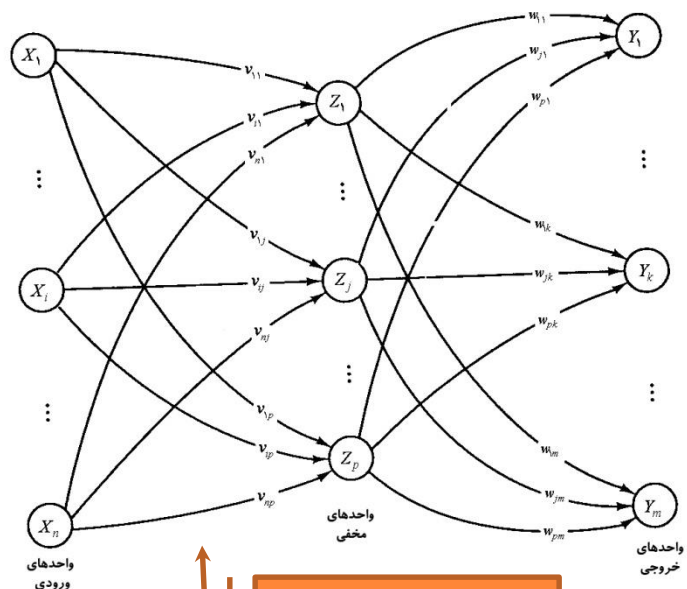
## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ ساختارهای رایج ...

- ساختار یا معماری: آرایش نرون‌ها در لایه‌ها و الگوهای ارتباط داخل و بین لایه‌ها
- شبکه‌های پیش‌خور (Feedforward) - سیگنال‌ها در یک جهت و از سمت واحدهای ورودی به سمت واحدهای خروجی (به سمت جلو) می‌روند



شبکه یک لایه



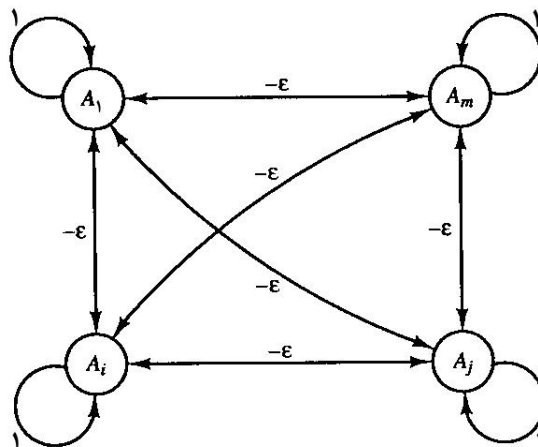
شبکه دولایه



## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ ساختارهای رایج

- شبکه بازگشتی (Recurrent)، مسیرهای بسته سیگنال از یک واحد به خودش وجود دارد
- شبکه رقابتی: واحدهای آن کاملاً به هم مرتبطند





## پیوند الگو - طبقه‌بندی الگو

### ○ بازشناسی الگو (Pattern Recognition)

- پیوند الگو (Pattern Association)

- پیوند دادن الگوی ورودی با یک الگوی خروجی

- ورودی: تصویر چهره یک فرد - خروجی: مشخصات و خصوصیات وی

- دسته‌بندی یا طبقه‌بندی الگو (Pattern Classification)

- حالت ساده (دو دسته): هر الگوی ورودی (یک بردار) عضو یک دسته است یا نه

- حالت کلی (n دسته): هر الگو (بردار ورودی)، به یکی از n دسته تعلق دارد





## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ تنظیم وزن‌ها ...

- آموزش: تعیین مقادیر وزن‌های شبکه

### • آموزش با نظارت (Supervised Learning)

- به ازای هر بردار ورودی، یک بردار هدف یا خروجی معادل در دسترس است
- طبقه‌بندی الگوها: بردار ورودی به دسته خاصی تعلق دارد (خروجی: «بله» یا «خیر»)
- پیوند الگو: خروجی یک الگو
- حافظهٔ انجمنی (Associative Memory): شبکه‌ای که برای پیوند مجموعه‌ای از بردارهای ورودی با مجموعه‌ای از بردارهای خروجی مطابق با آن آموزش داده می‌شود
- حافظهٔ خود انجمنی (Autoassociative Memory): بردار خروجی با بردار ورودی یکسان است
- حافظهٔ دیگر انجمنی (Hetroassociative Memory): بردار خروجی متفاوت از بردار ورودی است
- شبکه‌های پرسپترون چندلایه، شبکه‌های حافظهٔ انجمنی پیش‌خور و بازگشتی، یادگیری چندی‌سازی برداری (LVQ) و انتشار متقابل (CounterPropagation)



## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ تنظیم وزن‌ها

#### • آموزش بدون نظارت (Unsupervised Learning)

- بردارهای ورودی مشابه (دارای بیشترین شباهت) به هم را در یک دسته گروه‌بندی می‌کنند
- خوشه‌بندی (Clustering) بردارهای ورودی
- نگاشت‌های خودسازمانده کوهونن و نظریهٔ نوسان وفقی

#### • شبکه‌های با وزن ثابت

- نیازی به فرآیند آموزش تکراری ندارند
- حل مسائل بهینه‌سازی با محدودیت
- ماشین بولتزمن (بدون یادگیری) و شبکهٔ هاپلید پیوسته

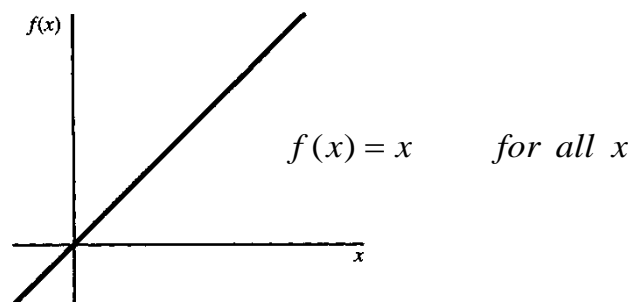


## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ توابع فعال‌سازی متداول ...

- تابع همانی (Identity Function)

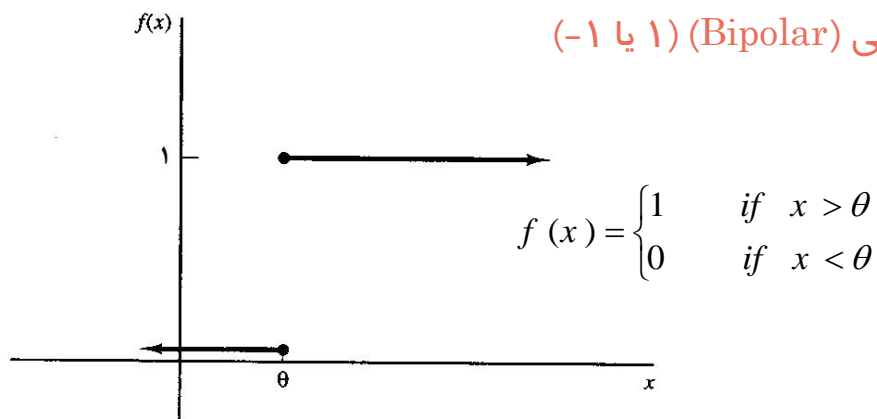
- برای واحدهای ورودی



- تابع پله‌ای دودویی (Step Function)

- تابع آستانه (Threshold Function) یا تابع هویساید (Heaviside Function)

- خروجی = سیگنال دودویی (۱ یا ۰) یا دوقطبی (Bipolar) (۱ یا -۱)



## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ توابع فعال‌سازی متداول

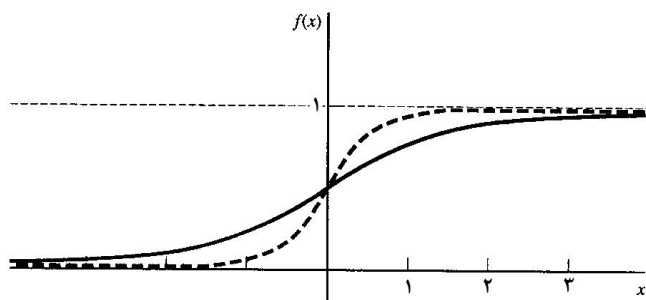
$$f(x) = \frac{1}{1 + \exp(-\sigma x)}$$

• توابع سیگموئید (Sigmoid Functions)

• منحنی‌هایی به شکل S

$$f'(x) = \sigma f(x)[1 - f(x)]$$

○ استفاده در شبکه‌های عصبی پس‌انتشار (نیاز به مشتق‌گیری)

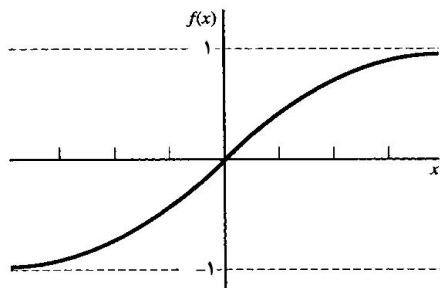


○ سیگموئید دودویی - تابع لجستیک (Logistic Function)

○ دامنه ۰ تا ۱، مقادیر مطلوب خروجی یا دودویی است و یا بین

○ سیگموئید دوقطبی - شبیه به تابع تانژانت هیپربولیک (Hyperbolic Tangent Function)

○ دامنه ۱- تا ۱



$$g(x) = 2f(x) - 1 = \frac{2}{1 + \exp(-\sigma x)} - 1 = \frac{1 - \exp(-\sigma x)}{1 + \exp(-\sigma x)}$$

$$g'(x) = \frac{\sigma}{2} [1 + g(x)][1 - g(x)]$$

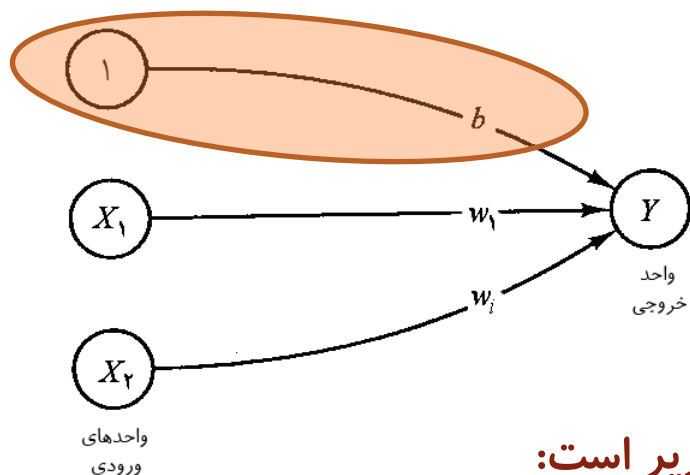




## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ بایاس ...

- در ورودی شبکه عصبی، علاوه بر ورودی‌های موردنظر، یک ورودی ثابت با مقدار ۱ نیز داشته باشیم.



$$y_{in} = 1 \times b + w_1 x_1 + w_2 x_2 = b + \sum_i w_i x_i$$

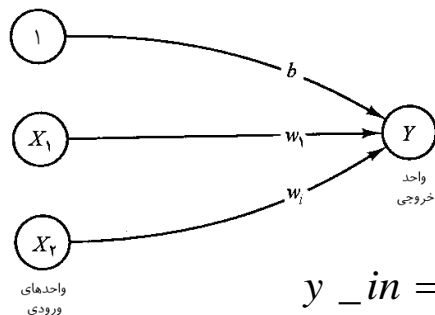
- تابع فعال‌سازی برای شبکه دارای بایاس به صورت زیر است:

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

## شبکه‌های عصبی مصنوعی: مفاهیم/تعاریف ...

### ○ بایاس ...

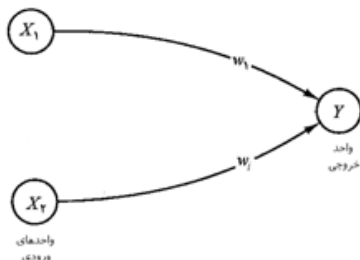
- وزن بایاس معادل آستانه ثابت در تابع فعال‌سازی پله



$$y_{in} = 1 \times b + w_1 x_1 + w_2 x_2 = b + \sum_i w_i x_i$$

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 0 \\ -1 & \text{if } y_{in} < 0 \end{cases} \Rightarrow b + w_1 x_1 + w_2 x_2 \geq 0 \Rightarrow w_1 x_1 + w_2 x_2 \geq -b$$

### ○ حالتی که بایاس نباشد ولی آستانه ثابت غیر صفر باشد



$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq \theta \\ -1 & \text{if } y_{in} < \theta \end{cases} \Rightarrow w_1 x_1 + w_2 x_2 \geq \theta$$

$$y_{in} = w_1 x_1 + w_2 x_2$$



## شبکه مک کلاچ-پیتز ...

### ○ نرون مک کلاچ-پیتز = اولین نرون مصنوعی

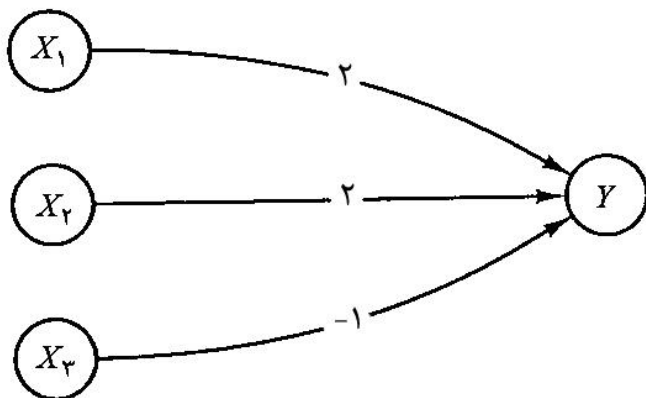
### ○ ویژگی‌ها

- تابع فعال‌سازی دودویی است
  - در هر مرحله زمانی، نرون یا برانگیخته می‌شود (فعال‌سازی ۱) و یا برانگیخته نمی‌شود (فعال‌سازی ۰)
- نرون‌های مک کلاچ-پیتز از طریق اتصالات مستقیم و وزن‌دار به هم متصل می‌شوند
- وزن مثبت روی اتصال = مسیر اتصال تحریکی، در غیر این صورت مسیر بازدارنده است.
  - تمام اتصالات تحریکی به یک نرون خاص، وزن‌های یکسان دارند
- هر نرون دارای سطح آستانه ثابتی است - اگر ورودی شبکه به آن نرون، بزرگ‌تر از مقدار آستانه باشد، نرون برانگیخته می‌شود
- سطح آستانه هر نرون طوری تعیین می‌گردد که بازدارندگی آن کامل باشد
  - ورودی بازدارنده غیر صفر مانع از برانگیخته شدن نرون می‌شود.
- عبور یک سیگنال از یک مسیر اتصال، به اندازه یک واحد زمانی طول می‌کشد

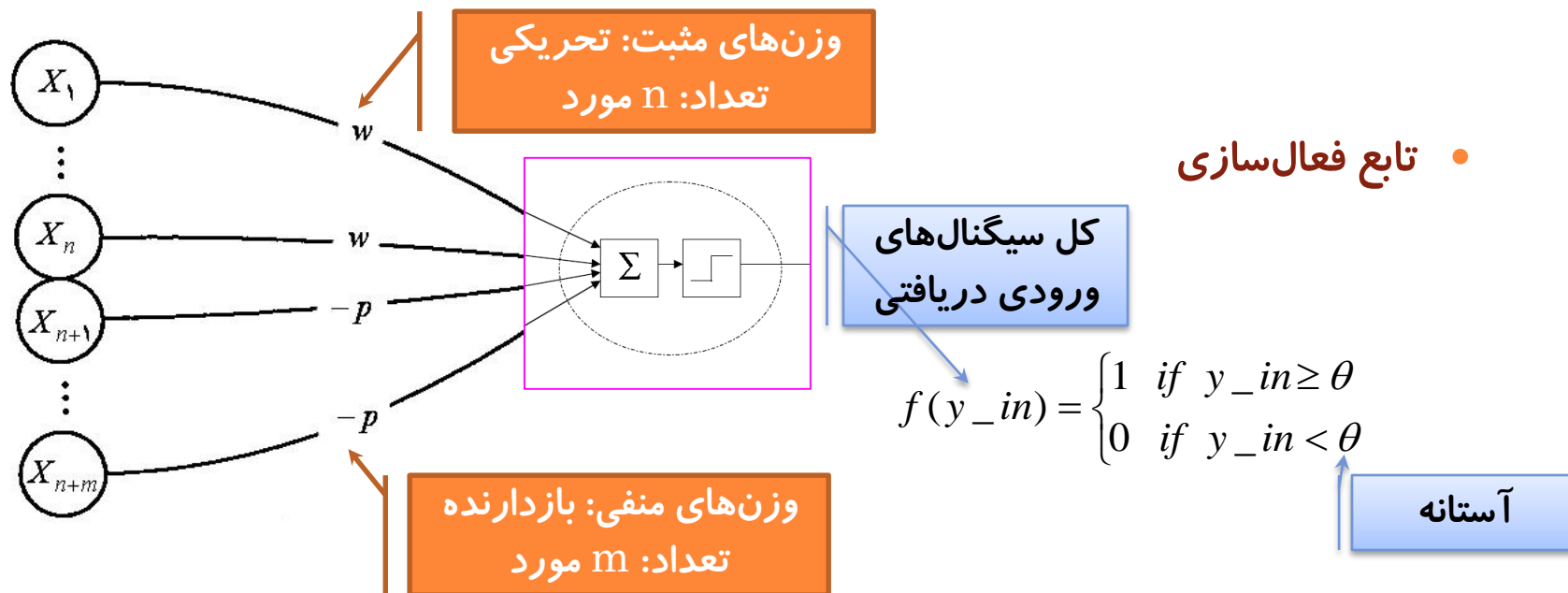
## شبکه مک کلاچ-پیتز...

### ○ مثال (نرون مک کلاچ-پیتز)

- اتصال از  $X_1$  به  $Y$  و اتصال از  $X_2$  به  $Y$ ، تحریکی هستند
  - اتصالات تحریکی وزن‌های مثبت یکسانی دارند، چون به یک واحد وارد می‌شوند
  - همچنین اتصال  $X_3$  به  $Y$ ، بازدارنده (منفی) است
  - یک واحد زمانی طول می‌کشد تا سیگنال از واحدهای  $X$  به واحد  $Y$  برسند
  - مقدار آستانه برای واحد  $Y$  برابر با ۴ است. چرا؟
- این مقدار به این واحد امکان می‌دهد که گاهی اوقات برانگیخته شود،
- اگر سیگنال غیرصفری در اتصال بازدارنده دریافت شود، از برانگیخته شدن واحد  $Y$  جلوگیری می‌کند



## شبکه مک کلاچ-پیتز: ساختار ...



تمام وزن‌های تحریکی که به هر واحد وارد می‌شوند باید یکسان باشد

بازدارندگی کامل = هر ورودی بازدارنده غیر صفر مانع از برانگیخته شدن نرون؟

$$\theta > nw - p$$



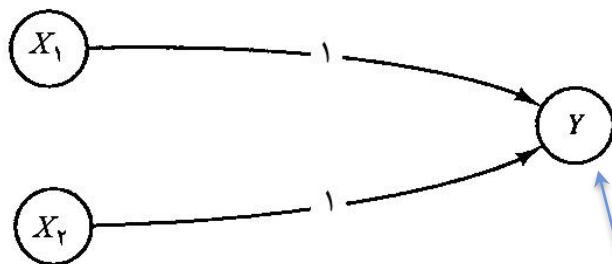
## شبکه مک کلاچ-پیتز: الگوریتم ...

○ وزن ثابت و مقدار آستانه مشخص برای تابع فعال‌سازی

• جهت تعریف توابع منطقی ساده

○ مثال: تابع AND

• خروجی «درست» است اگر هر دو مقدار ورودی «درست» باشند



$x_1$	$x_2$	$\rightarrow$	$y$
1	1		1
1	0		0
0	1		0
0	0		0

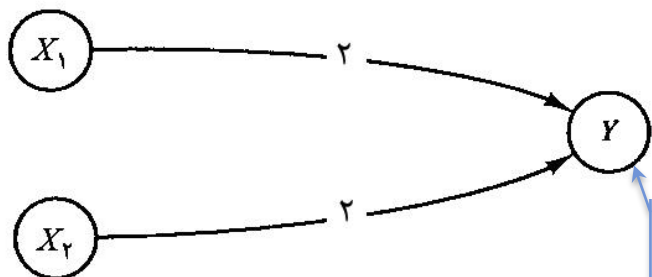
مقدار آستانه = ۲



## شبکه مک کلاچ-پیتز: الگوریتم ...

### مثال: تابع OR

- خروجی «درست» است اگر هر یک از مقادیر ورودی «درست» باشد

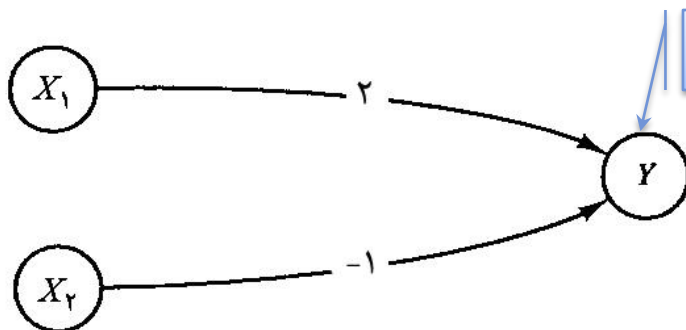


مقدار آستانه = ۲

$x_1$	$x_2$	$\rightarrow y$
1	1	1
1	0	1
0	1	1
0	0	0

### مثال: تابع AND NOT

- خروجی «درست» است اگر مقدار ورودی اول «درست» و مقدار ورودی دوم «نادرست» باشد



مقدار آستانه = ۲

$x_1$	$x_2$	$\rightarrow y$
1	1	0
1	0	1
0	1	0
0	0	0





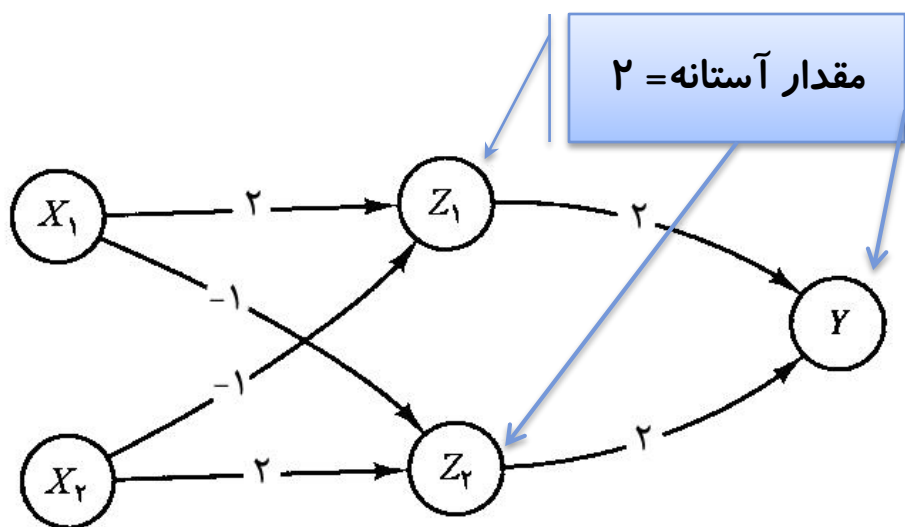
## شبکه مک کلاچ-پیتز: کاربرد

### ○ تابع XOR

- خروجی «درست» است اگر فقط یکی از مقادیر ورودی «درست» باشد

$x_1$	$x_2$	$\rightarrow$	$y$
1	1		0
1	0		1
0	1		1
0	0		0

$$x_1 \text{ XOR } x_2 \leftrightarrow (x_1 \text{ ANDNOT } x_2) \text{ OR } (x_2 \text{ ANDNOT } x_1)$$



- استفاده از یک شبکه دولایه

- لایه اول شامل دو عملگر AND NOT
- لایه دوم عملگر OR

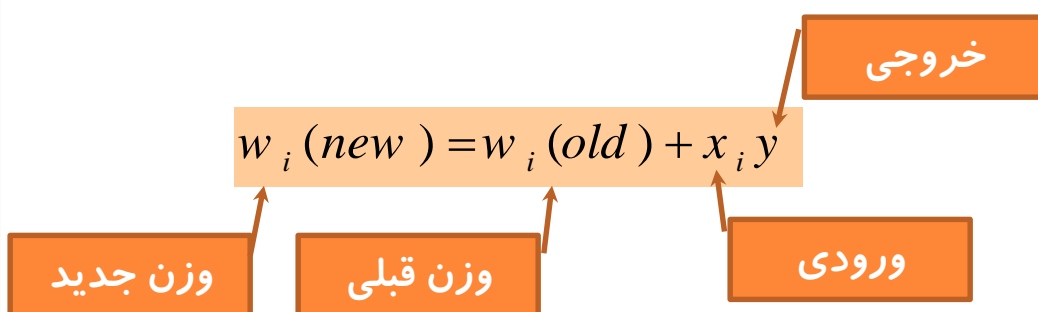


## شبکه هب ...

### اولین (و ساده‌ترین) قانون یادگیری برای شبکه عصبی

### ایده اصلی یادگیری هب

- یادگیری با تغییر استحکامات سیناپس‌های نرون‌ها (وزن‌های شبکه‌های عصبی) است
- اگر دو نرون متصل به هم به طور هم‌زمان «فعال» باشند، وزن بین آنها باید افزایش یابد
- هب درباره نرون‌هایی که به طور هم‌زمان برانگیخته نمی‌شوند، چیزی نمی‌گوید
- یادگیری قوی‌تر = اگر دو نرون به طور هم‌زمان «غیرفعال» باشند، وزن‌ها افزایش یابد



- شبکه هب یک لایه است
- به‌روز شدن (Update) وزن‌ها

• برای داده دودویی، اگر ورودی یا خروجی (یا هر دو) «غیرفعال» باشند، یادگیری صورت نمی‌گیرد



## شبکه هب: الگوریتم ...

- مرحله ۰ - به تمام وزن‌ها مقدار اولیه صفر بدهید  $w_i = 0 \quad (i = 1, \dots, n)$
- مرحله ۱ - برای هر بردار آموزش ورودی و خروجی هدف،  $s:t$ ، مراحل ۲ تا ۴ را انجام بده
- مرحله ۲ - فعال‌سازی‌های واحدهای ورودی را تعیین کن  $x_i = s_i \quad (i = 1, \dots, n)$
- مرحله ۳ - برای واحد خروجی فعال‌سازی را تعیین کن  $y = t$
- مرحله ۴ - وزن‌ها و بایاس را به‌روز کن

$$w_i(\text{new}) = w_i(\text{old}) + x_i y \quad (i = 1, \dots, n)$$

$$b(\text{new}) = b(\text{old}) + y$$

$$\mathbf{w}(\text{new}) = \mathbf{w}(\text{old}) + \mathbf{x} \cdot y \quad \Delta w = \mathbf{x} \cdot y \quad \Rightarrow \quad \mathbf{w}(\text{new}) = \mathbf{w}(\text{old}) + \Delta \mathbf{w}$$

داده‌های آموزشی فقط یک بار به شبکه نشان داده شده و آموزش به اتمام می‌رسد



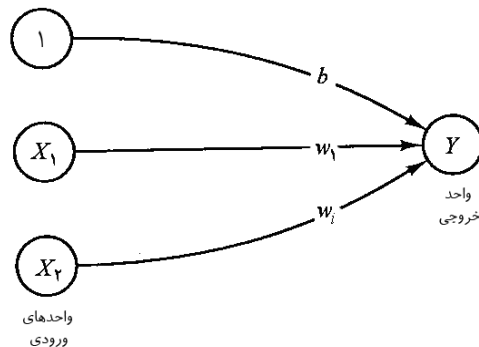
درس: مبانی رایانش نرم - شبکه‌های عصبی (مقدمه، شبکه مک کلاچ-پیتز و شبکه هب)

## شبکه هب: کاربرد ...

INPUT	TARGET
$(x_1 \quad x_2 \quad 1)$	
(1 1 1)	1
(1 0 1)	0
(0 1 1)	0
(0 0 1)	0

○ تابع AND با ورودی‌ها و هدف‌های دودویی ...

• تغییر وزن



$$\Delta w_1 = x_1 t, \quad \Delta w_2 = x_2 t, \quad \Delta b = 1 \cdot t = t$$

$$\mathbf{w}(\text{new}) = \mathbf{w}(\text{old}) + \Delta \mathbf{w}$$

$$x_1 = 1, \quad x_2 = 1, \quad b = 1, \quad t = 1$$

• برای ورودی اول

INPUT	TARGET	WEIGHT CHANGES	WEIGHTS
$(x_1 \quad x_2 \quad 1)$	$t$	$(\Delta w_1 \quad \Delta w_2 \quad \Delta b)$	$(w_1 \quad w_2 \quad b)$
(1 1 1)	1	(1 1 1)	(0 0 0)
(1 1 1)	1	(1 1 1)	(1 1 1)

مقدار اولیه

$$x_2 = -x_1 - 1$$



## شبکه هب: کاربرد ...

### ○ تابع AND با ورودی‌ها و هدف‌های دودویی

• برای دومین، سومین و چهارمین ورودی

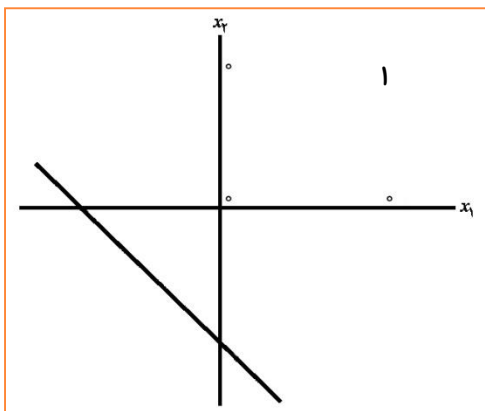
INPUT	TARGET	WEIGHTCHANGES	WEIGHTS
$(x_1 \ x_2 \ 1)$	$t$	$(\Delta w_1 \ \Delta w_2 \ b)$	$(w_1 \ w_2 \ b)$
$(1 \ 0 \ 1)$	0	$(0 \ 0 \ 0)$	$(1 \ 1 \ 1)$
$(0 \ 1 \ 1)$	0	$(0 \ 0 \ 0)$	$(1 \ 1 \ 1)$
$(0 \ 0 \ 1)$	0	$(0 \ 0 \ 0)$	$(1 \ 1 \ 1)$

یادگیری رخ نمی‌دهد  
وزن‌ها تغییر نمی‌کند

الگوهایی با مقدار هدف صفر یا  
«غیر فعال»

استفاده از نمایش دودویی

پاسخ  
نادرست



$$x_2 = -x_1 - 1$$



## شبکه هب: کاربرد ...

### ○ تابع AND با ورودی‌های دودویی و مقادیر هدف دوقطبی

#### • اولین ورودی

INPUT ( $x_1$ $x_2$ 1)	TARGET $t$	INPUT ( $x_1$ $x_2$ 1)	TARGET $t$	WEIGHT CHANGES ( $\Delta w_1$ $\Delta w_2$ $\Delta b$ )	WEIGHTS ( $w_1$ $w_2$ $b$ )
(1 1 1)	1	( $x_1$ $x_2$ 1)	$t$	( $\Delta w_1$ $\Delta w_2$ $\Delta b$ )	( $w_1$ $w_2$ $b$ )
(1 0 1)	-1				(0 0 0)
(0 1 1)	-1	(1 1 1)	1	(1 1 1)	(1 1 1)
(0 0 1)	-1				

$$x_2 = -x_1 - 1$$

#### • ارائه دومین، سومین و چهارمین

INPUT ( $x_1$ $x_2$ 1)	TARGET $t$	WEIGHT CHANGES ( $\Delta w_1$ $\Delta w_2$ $b$ )	WEIGHTS ( $w_1$ $w_2$ $b$ )
(1 0 1)	-1	(-1 0 -1)	(0 1 0)
(0 1 1)	-1	(0 -1 -1)	(0 0 -1)
(0 0 1)	-1	(0 0 -1)	(0 0 -2)

پاسخ  
نادرست





## شبکه هب: کاربرد ...

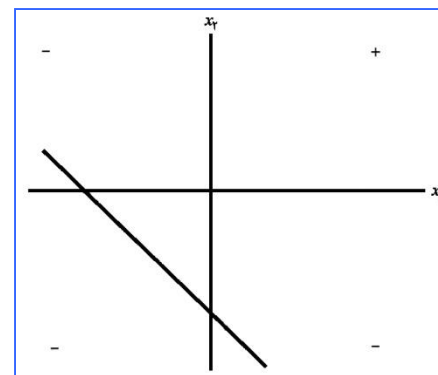
INPUT	TARGET
$(x_1 \ x_2 \ 1)$	$t$
$(1 \ 1 \ 1)$	1
$(1 \ -1 \ 1)$	-1
$(-1 \ 1 \ 1)$	-1
$(-1 \ -1 \ 1)$	-1

○ تابع AND برای ورودی‌ها و هدف‌های دوقطبی ...

• اولین ورودی

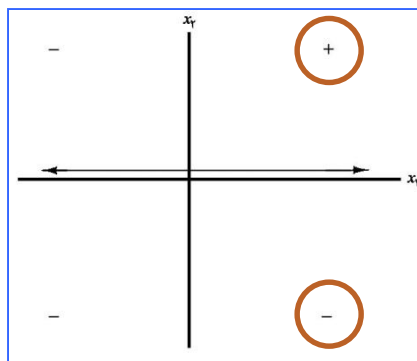
INPUT	TARGET	WEIGHT CHANGES	WEIGHTS
$(x_1 \ x_2 \ 1)$	$t$	$(\Delta w_1 \ \Delta w_2 \ \Delta b)$	$(w_1 \ w_2 \ b)$
			$(0 \ 0 \ 0)$
$(1 \ 1 \ 1)$	1	$(1 \ 1 \ 1)$	$(1 \ 1 \ 1)$

$$x_2 = -x_1 - 1$$



INPUT	TARGET	WEIGHT CHANGES	WEIGHTS
$(x_1 \ x_2 \ 1)$	$t$	$(\Delta w_1 \ \Delta w_2 \ \Delta b)$	$(w_1 \ w_2 \ b)$
			$(1 \ 1 \ 1)$
$(1 \ -1 \ 1)$	-1	$(-1 \ 1 \ -1)$	$(0 \ 2 \ 0)$

$$x_2 = 0$$



• دومین ورودی

پاسخ درست برای  
دو نمونه آموزش



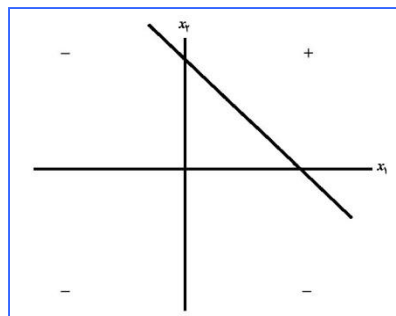
## شبکه هب: کاربرد ...

### تابع AND برای ورودی‌ها و هدف‌های دوقطبی

• سومین ورودی

INPUT	TARGET	WEIGHT CHANGES	WEIGHTS
$(x_1 \ x_2 \ 1)$	$t$	$(\Delta w_1 \ \Delta w_2 \ \Delta b)$	$(w_1 \ w_2 \ b)$
$(0 \ 2 \ 1)$			$(0 \ 2 \ 0)$
$(-1 \ 1 \ 1)$	-1	$(1 \ -1 \ -1)$	$(1 \ 1 \ -1)$

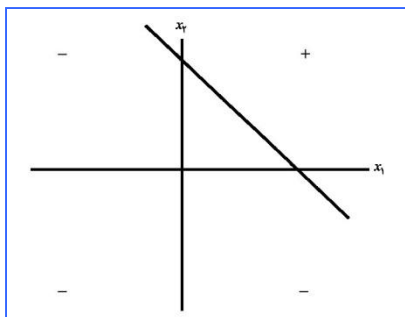
$$x_2 = -x_1 + 1$$



INPUT	TARGET	WEIGHT CHANGES	WEIGHTS
$(x_1 \ x_2 \ 1)$	$t$	$(\Delta w_1 \ \Delta w_2 \ \Delta b)$	$(w_1 \ w_2 \ b)$
$(1 \ 1 \ 1)$			$(1 \ 1 \ -1)$
$(-1 \ -1 \ 1)$	-1	$(1 \ 1 \ -1)$	$(2 \ 2 \ -2)$

• چهارمین ورودی

$$x_2 = -x_1 + 1$$







## شبکه هب: نمایش داده‌ها

○ شکل نمایش داده‌ها می‌تواند مسئله قابل حل را به مسئله‌ای غیرقابل حل تبدیل کند

• در قانون هب بسیار موثر است

○ برای برخی الگوها منجر به جواب درست نمی‌شود، ممکن است برای نمایش متفاوتی از همان الگوها پاسخ درستی را نتیجه دهد



○ نمایش دوقطبی بهتر از نمایش دودویی است

• افزایش قابلیت تعمیم شبکه

• امکان تمایز داده‌های گم‌شده (Missing Data) از داده‌های اشتباه (Mistaken Data)

○ مقادیر گم‌شده = «۰»

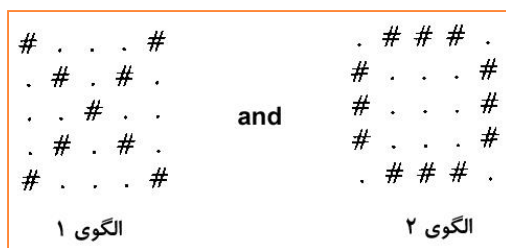
○ اشتباهات = قرینه مقدار ورودی از  $+1$  به  $-1$ ، یا برعکس



## شبکه هب: مثال ...

### ○ بازشناسی نویسه (کاراکتر) - الگوهای ورودی دوبعدی ...

- یک شبکه هب برای تشخیص الگوی «X» از الگوی «O»

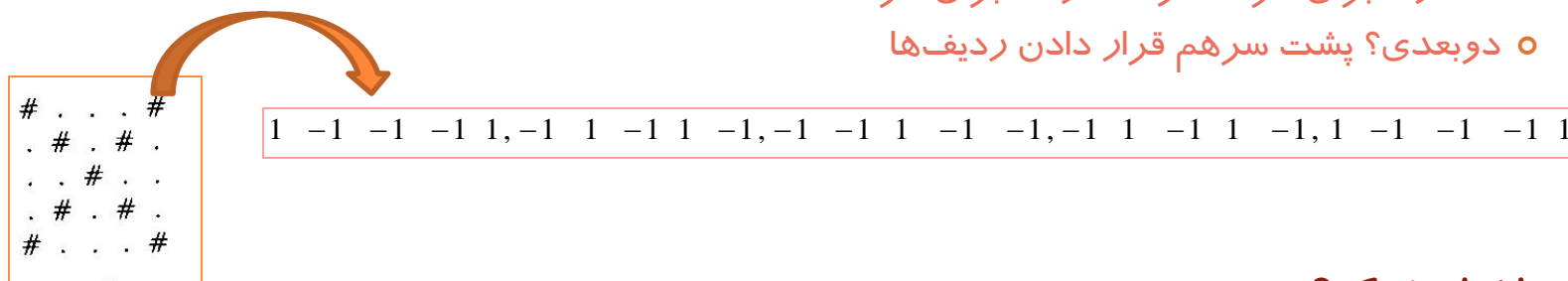


- یک مسئله دسته‌بندی الگو؟

- دسته «X» = خروجی مورد نظر و الگوی «O» = خروجی «غیر X»
- روش دیگر؟

- تبدیل الگوهای «O» و «X» به بردارهای ورودی؟

- مقدار ۱ برای هر «#» و مقدار -۱ برای هر «۰»
- دوبعدی؟ پشت سرهم قرار دادن ردیف‌ها



- ساختار شبکه؟

- تعداد نرون‌های ورودی = برابر با تعداد ابعاد بردار الگوها = ۲۵



## شبکه هب: مثال ...

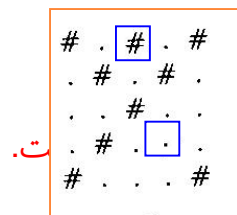
### ○ بازشناسی نویسه (کاراکتر) - الگوهای ورودی دوبعدی ...

#### • قابلیت تعمیم شبکه

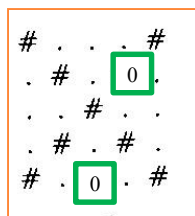
○ تولید پاسخ منطقی شبکه برای الگوهای ورودی شبیه الگوهای آموزش اما نه کاملاً یکسان با آنها

#### • دو نوع تغییر در الگوی ورودی

○ «اشتباهات در داده‌ها»



○ علامت یک یا چند مؤلفه بردار ورودی قرینه شده و از ۱ به -۱، یا برعکس،



○ «داده‌های گم‌شده»

○ یک یا چند مؤلفه بردار ورودی به جای مقدار ۱ یا -۱ مقدار صفر دارند

### • شبکه در برخورد با داده‌های گم‌شده عملکرد بهتری در مقایسه با اشتباهات دارد

○ در مورد داده‌های ورودی، «بهتر است که حدس نزنیم»!!



## جداسازی خطی ...

### ○ مسئله دسته‌بندی ساده با شبکه عصبی

- الگوی ورودی عضو دسته مورد نظر باشد، پاسخ «بله» و اگر ورودی عضو آن دسته نباشد، پاسخ «خیر»

$$y_{in} = b + \sum_i x_i w_i$$

- تابع فعال‌سازی پله‌ای

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq 0 \\ -1 & \text{if } y_{in} < 0 \end{cases}$$

- مرز تصمیم‌گیری (Decision Boundary) = مرز بین ناحیه‌ای که در آن  $y_{in} > 0$  و ناحیه‌ای که در آن  $y_{in} < 0$  است

$$b + \sum_i x_i w_i = 0$$

- پاسخ این معادله یک خط، یک صفحه و یا یک ابرصفحه است

- وابسته به تعداد واحدهای ورودی (ابعاد بردار ورودی)



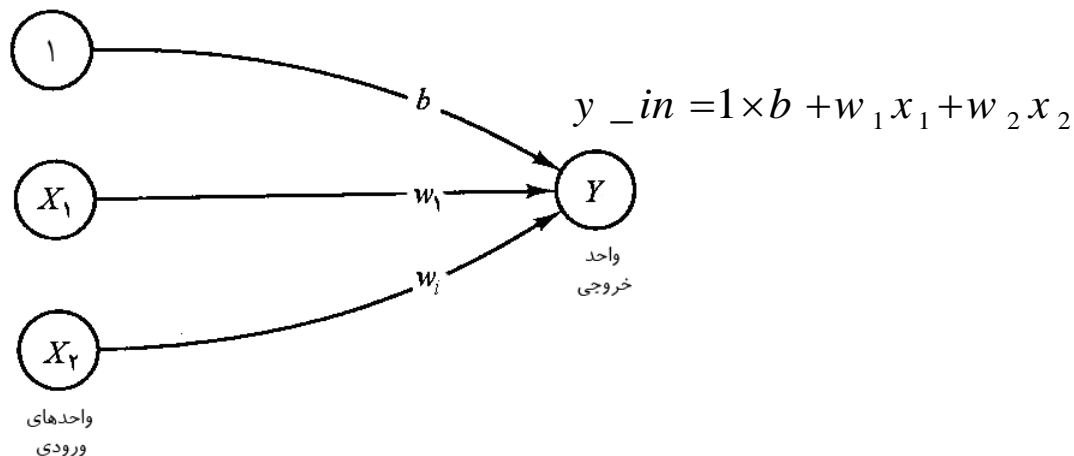
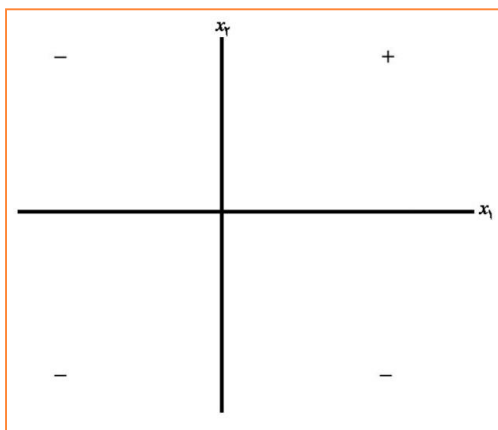
## جداسازی خطی ...

### ○ مسئله خطی تفکیک‌پذیر (Linearly Separable)

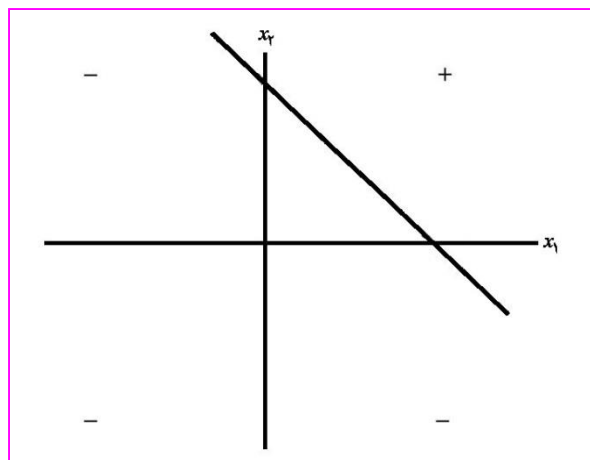
- حل یک مسئله توسط شبکه‌های یک لایه پس از تنظیم وزن‌ها (و بایاس)
- تمام بردارهای ورودی آموزش که پاسخ صحیح برای آنها  $+1$  (عضویت در دسته) است، در یک طرف مرز تصمیم‌گیری و تمام بردارهای ورودی آموزش که پاسخ صحیح برای آنها  $-1$  (عدم عضویت در دسته) است در سمت دیگر مرز تصمیم‌گیری قرار می‌گیرند
- نشان داده شده است که شبکه‌ی یک لایه فقط می‌تواند مسائلی را حل کند که به صورت خطی تفکیک‌پذیر باشند
- شبکه‌های چندلایه‌ای که از توابع فعال‌سازی خطی استفاده می‌کنند، از شبکه‌های یک لایه قوی‌تر نیستند زیرا ترکیب چند تابع خطی نیز خطی است

## جداسازی خطی ...

INPUT( $x_1, x_2$ )	OUTPUT
(1, 1)	+1
(1, -1)	-1
(-1, 1)	-1
(-1, -1)	-1



○ مثال: تابع AND



• مرز تصمیم‌گیری  $b + w_1 x_1 + w_2 x_2 = 0$

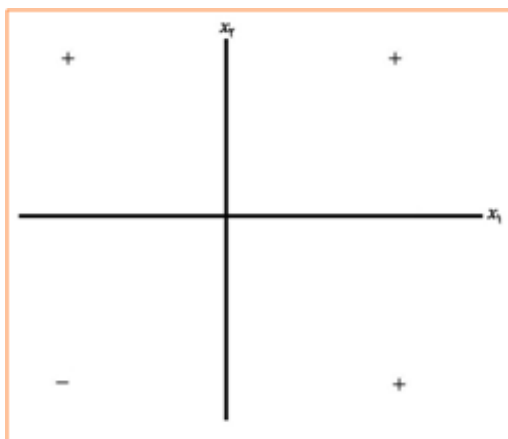
• پاسخ  $b = -1, w_1 = 1, w_2 = 1$

$x_2 = -x_1 + 1$

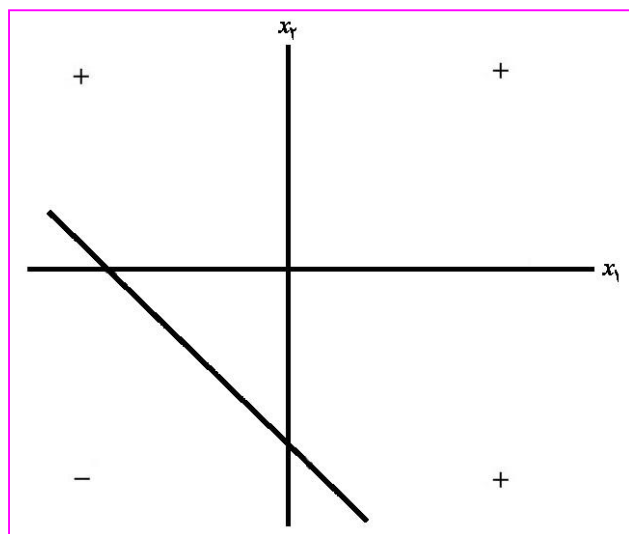


## جداسازی خطی ...

○ مثال: تابع OR



INPUT( $x_1, x_2$ )	OUTPUT
(1, 1)	+1
(1, -1)	+1
(-1, 1)	+1
(-1, -1)	-1



• مرز تصمیم‌گیری

$$b = 1, w_1 = 1, w_2 = 1$$

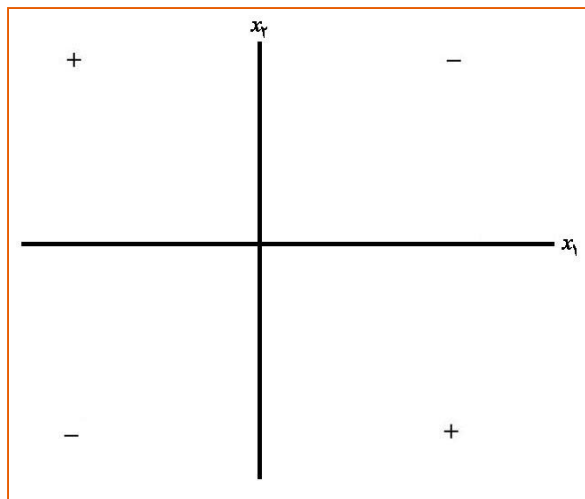
$$x_2 = -x_1 - 1$$

• اگر وزن بایاس وجود نداشت، مرز تصمیم‌گیری باید از مبدأ عبور می‌کرد



## جداسازی خطی

### مثال: تابع XOR



INPUT( $x_1, x_2$ )	OUTPUT
(1, 1)	-1
(1, -1)	+1
(-1, 1)	+1
(-1, -1)	-1

- حل؟
- فضای داده‌های ورودی به صورت خطی جدایی پذیر نیست.
- هیچ خط مستقیم نمی‌تواند نقاط مثبت و منفی را جدا کند