کاربرد هوش مصنوعی در IT

امروزه از هوش مصنوعی در تجارت الکترونیک ، گردشگری ، پزشکی و صنعت استفاده می‌شود.

الف - نقش AI در تجارت الکترونیکی

استفاده از AI در تجارت B2B و C2C روز به روز گسترش می‌یابد:

برای مثال در تجارت B2C از AI، بیشتر برای انتخاب و پیشنهاد محصول به مشتری ، پاسخگویی به سؤالات مشتری و دسته‌بندی و قیمت‌گذاری کالا استفاده می‌شود. سیستم‌های مشاوره هوشمند ، نقش مهمی را در اینترنت و انبوه اطلاعات ارایه شده در آن برعهده دارد.

برای مثلا سیستم‌های AFC (فیلتر خودکار تعاملی) که با توجه به اولویت‌های مصرف‌کننده قبلی ، الگوهای خرید جدیدی را پیشنهاد می‌کند (Groplent) و یا سیستم‌های دانش‌محور (KB Base) که براساس دانش از محصول ، کاربران را هدایت می‌کند. (Case Base Reasoning)

نکته مهم: امروزه مشتریان در فضای اینترنت با مشاهده متن‌های توضیحی (Text hasted) یا تعامل با بخش‌های پرس و جو گرا (Query Based) راضی نمی‌شوند. بنابراین روش‌های مزیت‌گرا (Preference Based) مبتنی بر AI به سرعت در حال جایگزینی با روش‌های قدیمی می‌باشد.

۱- مثال‌هایی از AI در B2B

در تجارت AI به طور وسیع در مدیریت حلقه تأمین (SCM) استفاده می‌شود. مدیریت حلقه تأمین وظیفه تأمین به موقع (نه زود و نه دیر) قطعات از قطعه‌سازان را برای تولیدی به موقع و ارزان برعهده دارد. سفارش زودتر از موعد باعث افزایش هزینه انبارداری و تأخیر در سفارش باعث تأخیر در تولید و هدررفتن سرمایه می‌شود.

۲- نقش AI در صنعت و گردشگری

الف) گردشگری الکترونیکی (E- Turism): امروزه مسافران از طریق اینترنت به دنبال آژانس‌های مسافرتی خوبی هستند که بتوانند قیمت‌های مناسب و کیفیت بالایی را ارایه دهند. عواملی چون

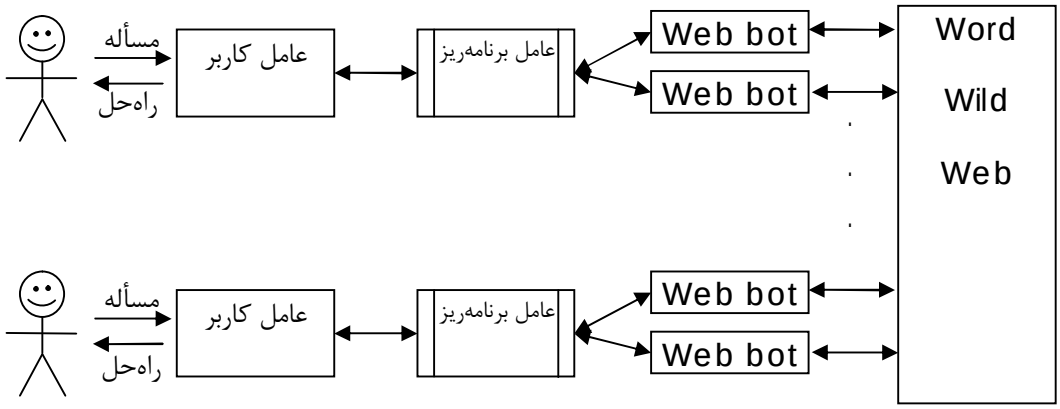
سفر ، انتخاب وسیله سفر ، نوع و قیمت هتل و ... همچنین اولویت‌های مسافر ، بسیار مهم می‌باشند.

به دست آوردن بهترین روش سفر برای یک مسافر خاص از نظر هزینه و ... . بنابراین طراحی سیستمی که بتواند برای یک مسأله راه‌حل‌های مختلفی را ارایه دهد و این راه‌حل‌ها بر اساس هزینه ، زمان و اولویت‌های کاربر بهینه باشد ، اساس رقابت بین آژانس‌های مسافرتی خواهد بود. به همین دلیل امروزه از سیستم‌های چندعاملی MAS (Multi Agent Systems) که زیرگروهی از هوش مصنوعی توزیع شده است در E- Tourism استفاده می‌شود. برخی از این عوامل عبارتند از:

۲-۱- عامل کاربر: سؤالات کاربر را دریافت کرده و پس از تجزیه و تحلیل مسأله آن‌را برای عامل برنامه‌ریز ارسال می‌کند.

۲-۲- عامل برنامه‌ریز: این عامل با استفاده از فنون استدلال و تعامل با سایر عوامل و بر اساس تکنیک‌های CBR برای فهرست و طبقه‌بندی موضوع عمل می‌کند.

۲-۳- ماشین‌های Web bot: این عامل اطلاعات مورد نیاز برای عامل برنامه‌ریز را از طریق اینترنت فراهم می‌کند.



MAS در E- Tourism

ب - نقش AI در صنعت

با وجود آنکه تحقیق در شبکه‌های عصبی زودتر از سیستم‌های خبره آغاز شده بود ولی اولین موفقیت‌های AI در زمینه دانش ماشین نمادین مربوط به اواخر دهه ۶۰ و اوایل دهه ۷۰ بود. برای مثال یکی از کاربردهای سیستم خبره My CIN بود که برای تشخیص بیماری‌های عفونت خونی و در دانشگاه استنفورد در سال ۱۹۷۲ به وجود آمد. اولین کاربرد تجاری سیستم خبره XCONبود که توسط شرکت DEC و برای پیکره‌بندی سیستم‌های کامپیوتری VAX به کار رفت. منطق Fuzzy نیز که توسط پروفسور لطفی‌زاده ارایه شده بود و به بررسی رفتار مبهم یا غیرصریح عناصر و مفاهیم موجود در دنیا می‌پرداخت موفق شد تا برای اولین تجربه تجاری در سال ۱۹۸۹ توسط شرکت ماتسوشیتا و ماشین‌ظرفشویی پیاده‌سازی و فروش بسیار خوبی داشته باشد. اگرچه در شرکت هیتاچی از این منطق برای کنترل راه‌آهن ژاپن استفاده شده بود.

یکی دیگر از پیشرفت‌های مهم در AI الگوریتم ژنتیک GA می‌باشد که با الهام از طبیعت (ژن‌ها و کروموزوم‌ها) به جستجو و یافتن راه‌حل‌های بهینه می‌پردازد. از این الگوریتم در طراحی مدارهای VLSI ، بازی ، شناسایی چهره و تقریبا در همه علوم استفاده می‌شود.

امروزه تکنولوژی‌های هوش مصنوعی با یکدیگر ترکیب شده و مدل‌های hybrid مختلفی را ایجاد کرده‌اند. از اواسط دهه ۱۹۹۰ تکنولوژی شبکه‌های عصبی با تکنولوژی فازی تلفیق شد و NeuroFuzzy در سیستم‌های هوشمند و رباط‌ها مورد استفاده قرار گرفت. مثلا نرم‌افزار Process In Sights که با استفاده از NeuroFuzzy پیاده‌سازی شد و برای مدل کردن و بهینه‌سازی فرآیندها استفاده می‌شود. نرم‌افزار توانسته است سالانه بیش از یک میلیون دلار صرفه‌جویی داشته باشد. همچنین سرمایه‌گذاری کشورهای پیشرو در صنعت رباتیک نشان از اهمیت این موضوع دارد. برای مثال در سی سال گذشته ، دولت ژاپن ۴۰۰ میلیون دلار برای توسعه یک رباط انسان‌نما با ظرفیت هوشی ، جسمی و احساسی کمتر از یک انسان ۵ ساله هزینه کرده است. برخی از شرکت‌های مهم ژاپن که در زمینه رباط‌های انسان‌نما فعالیت می‌کنند و عبارتند از هوندا ، میتسوبیشی ، سونی ، فوجیتسو ، پاناسونیک و NEC.

تحقیق: پیاده‌سازی سخت‌افزاری و نرم‌افزاری ربات‌ها با چه نرم‌افزارهایی صورت می گیرد

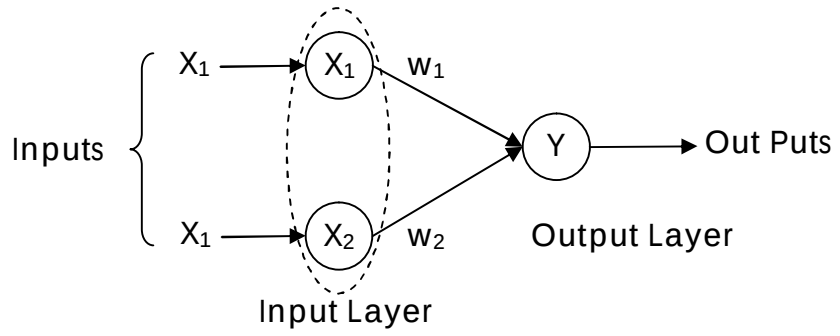
برای مثال ربات انسان‌نمای ASIMO شرکت هوندا از جمله این موارد می‌باشد.

شبکه‌های عصبی هوشمند یا ANN (Artificial Neural Networks)

ابزاری برای پردازش اطلاعات غیرخطی (Nonlinear) می‌باشند که از یک سری واحدهای پردازش به هم مرتبط به نام Neuron تشکیل شده است. در حقیقت شبکه‌های عصبی هوشمند نوعی از هوش مصنوعی است که تلاش می‌کند عملکرد مغز انسان را تقلید کند به همین دلیل می‌توان آن‌را مجموعه‌ای از پردازنده‌های توزیع‌شده موازی (Parallel Distributed Processors) در نظر گرفت که توانایی ذخیره دانش را دارند. به طور خلاصه شبکه‌های عصبی را می‌توان یک سیستم پردازش اطلاعات و یا یک الگوریتم محاسبات غیرخطی برای پردازش داده ، سیگنال و تصویر دانست که این عمل توسط نرون‌ها انجام می‌شود. هر نرون هوشمند دارای سه مشخصه زیر است:

۱- معماری و ساختار (نحوه ارتباط نرون‌ها با هم)
۲- آموزش و یادگیری (تعیین وزن رابطها)
۳- تابع فعال‌ساز (Activation Function)

شکل زیر یک شبکه عصبی هوشمند ساده را نشان می‌دهد:



دلایل استفاده از ANN:

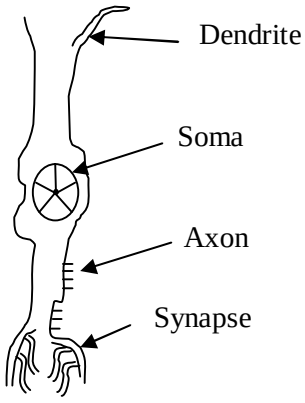
۱- توانایی یادگیری و تعمیم به موارد مشابه
۲- موازی‌سازی و توانایی نمایش Representation و محاسبه توزیعی Distribute
۳- توانایی انطباق با تجربیات گذشته
۴- تحمل خطا یا Fault Tolerant و مصرف انرژی پایین

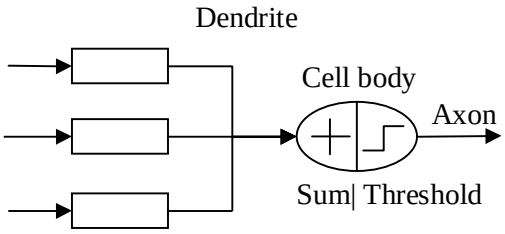
ساختار سلول‌های عصبی انسان

یک سلول عصبی شامل بخش‌های زیر می‌باشد:

۱- Dendrite: وظیفه دریافت سیگنال از سایر نرون ها را بر عهده دارد.
۲- Soma: تمام سیگنال‌های ورودی را با هم جمع می‌کند.
۳- Axon: پس از دریافت مقدار مشخصی ورودی سلول تحریک شده و سیگنالی را از طریق Axon و سیناپس (Synaps) به سایر سلول‌ها منتقل می‌کند.
۴- Synapse: ارتباط الکتروشیمیایی بین نرون‌ها را فراهم می‌کند.

شکل زیر شبکه عصبی طبیعی و مصنوعی را نمایش می‌دهد:





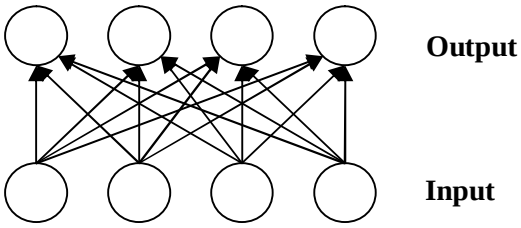
شبکه عصبی طبیعی	شبکه عصبی مصنوعی
Neuron	Cell body
Dendrite	Weight
Soma	Net Input
Axon	Output

تذکر: مغز انسان در حدود 10^{11} نرون و 10^{15} ارتباط دارد که می‌تواند عمل شناسایی یک الگوی پیچیده را (Pattern recognition) را در ۱۰۰ تا ۲۰۰ میلی‌ثانیه انجام دهد در حالی که یک شبکه عصبی مصنوعی ممکن است نیاز به ساعت‌ها پردازش داشته باشد.

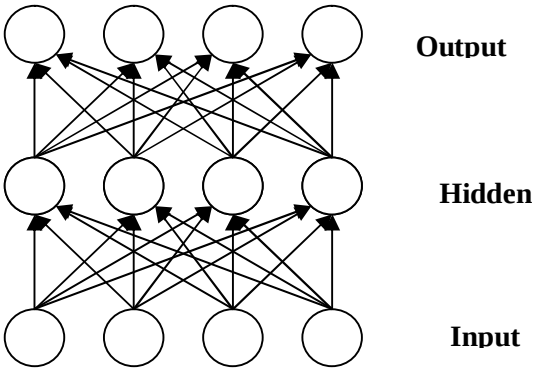
انواع شبکه عصبی از نظر ساختار

برخی از ساختارهای متداول در شبکه‌های عصبی به صورت زیر است:

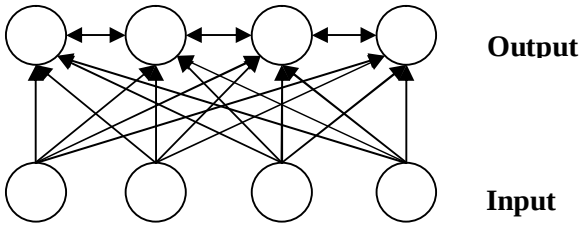
۱- Single layer feed forward



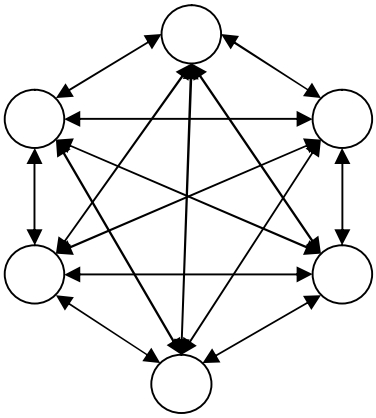
۲- Multi layer feed



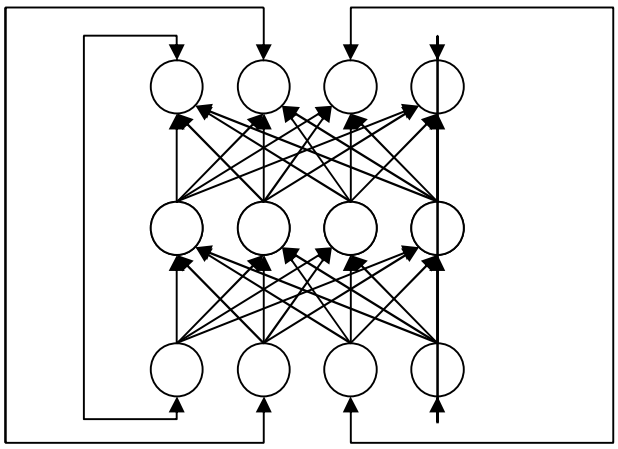
۳- Competitive network



٤- Fully Recurrent network



٥- Jordan Network



انواع شبکه‌های عصبی از لحاظ آموزش و یادگیری

به روش تعیین وزن‌های (Weights) مناسب برای شبکه‌های عصبی ، آموزش (training) و یادگیری می‌گویند که به سه دسته کلی تقسیم می‌شود:

۱- **Supervised**: در یان روش به ازای یک سری ورودی ، خروجی شبکه با پاسخ‌های مورد انتظار مقایسه می‌شود و آن‌قدر این مقایسه و تغییر مقادیر وزن تکرار می‌شوند تا به پاسخ‌های از قبل تعیین شده برسند. برای مثال شبکه‌های عصبی Hebbnet ، Pattern Associative Memory ، Perceptron – Madaline – Adaline ، Back Propagation

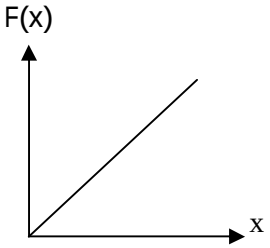
۲- **Unsupervised**: اگر در یک شبکه عصبی به ازای مقادیر ورودی (بردار ورودی) مقادیر خروجی هدف مشخص نباشد روش آموزش و یادگیری شبکه را Unsupervised می‌گویند. پیاده‌سازی این شبکه‌ها پیچیده و دارای فیدبک می‌باشند که به این شبکه‌ها Self learning (خودآموز) یا Self Organization نیز می‌گویند. فرآیند یافتن وزن مناسب آن‌قدر تکرار می‌شود تا به یک مقدار پایدار (Stable) همگرا شود. برای مثال شبکه‌های عصبی ART (Adaptive Resonance Theory) و SOM (Self Organization Map) از این روش استفاده می‌کند.

۳- **ReInforcement**: در این روش از یک ناظر یا معلم (teacher) برای شکبه استفاده می‌شود که با جواب درست یا غلط (سعی و خطا) شبکه را به سمت جواب نهایی سوق می‌دهد که این جواب نهایی در ابتدا مشخص نبود و در حقیقت نوع دیگری از روش Supervised می‌باشد.

تابع فعال‌ساز Activation Function

از این تابع برای محاسبه خروجی یک نرون استفاده می‌شود. برخی از مهم‌ترین توابع فعال‌ساز عبارتند از:

۱- تابع Identity

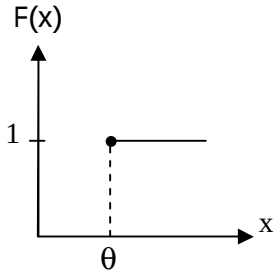


F(x)= x

۲- تابع Binary Step

$$f(x)=\begin{cases}1 & if \quad f(x)\geq q\\ 0 & if \quad f(x)<q\end{cases}$$

q : *Threshold*



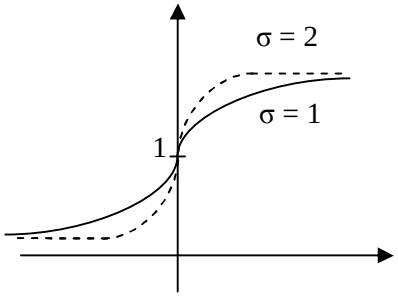
$$f(x)=\begin{cases}+1 & if \quad f(x)\geq q\\ -1 & if \quad f(x)<q\end{cases}$$

q : *Threshold*

۳- تابع Sigmoidal

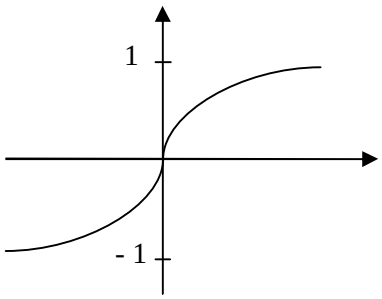
۳-۱- Binary Sigmoidal

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



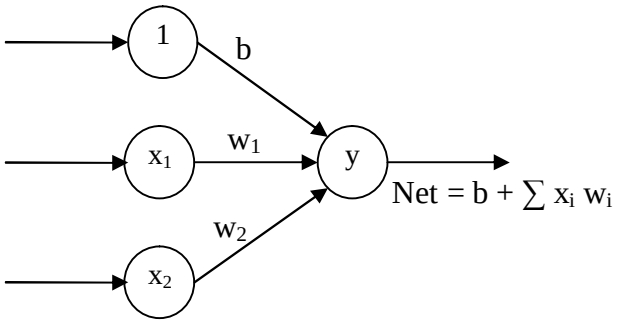
Bipolar Sigmodal -۲-۳

$$b(x) = 2 f(x) \qquad f(x) = \frac{1-\exp(-sx)}{1+\exp(-sx)}$$



مفهوم Bias

عملکرد Bias مشابه وزن در شبکه که دارای یک مقدار معین می‌باشد (یک یا صفر)



که باعث بهبود عملکرد شبکه می‌باشد. برای مثال شکل فوق را در نظر بگیریدکه مقدار خروجی شبکه با استفاده از یک Activation Function به دست می‌آید. برای مثال:

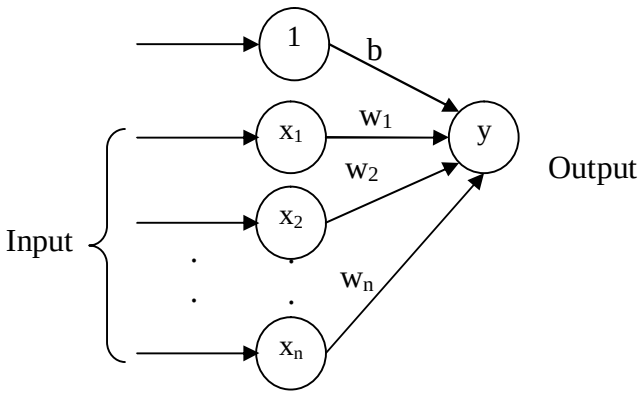
$$y = f(Net) = \begin{cases} -1 & if \quad Net \geq q \\ +1 & if \quad Net < q \end{cases}$$

$$q : Threshold$$

شبکه عصبی Heb

اولین قانون یادگیری شبکه‌های عصبی توسط Donal Heb (۱۹۴۹) طراحی شد. این قانون بعدها توسط Mc Clelland و Rumel Hurt (۱۹۸۸) توسعه یافت و به نام شبکه عصبی Heb مشهور شد.

مقادیر ورودی این شبکه باید به صورت دو قطبی (Bipolar) باشند. این شبکه از نوع Single Layer Feed Forward بوده که برای تشخیص و طبقه‌بندی الگو (Pattern Classification) از آن استفاده می‌شود. ساختار این شبکه به صورت زیر است:



نکته: تابع فعال‌ساز این شبکه از نوع Identity می‌باشد.

مراحل الگوریتم شبکه Heb:

۱- مقداردهی اولیه تمام وزن‌ها و Bios به $b= 0$ و $w_1 = 0$ $(1\leq i\leq n)$

۲- تعیین بردارهای ورودی $(x_i = S)$ و خروجی هدف $(y = t)$

۳- تنظیم وزن‌ها با استفاده از فرمول زیر (قانون Heb)

$\Delta w_i = x_i y$ " $w_i(new) = w_i(old) + x_i y$

۴- تنظیم bios با استفاده از فرمول زیر:

$\Delta b = y$ " $b(new) = b(old) + y$

مفهوم جداپذیری خطی (Linear Seperability)

طبق تعریف $y = b + \sum_i x_i w_i$ رابطه زیر را ناحیه مرزی (boundary region) می‌نامند:

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

که اگر بتوان مقادیر وزن بردارهای آموزش را با استفاده از یک مرز تفکیک کرد به آن جداپذیر خطی (Linear separable) و در غیر این صورت (linear non- separable) می‌گویند.

مثال: تعریف یک شبکه Heb برای تابع AND با ورودی‌ها و خروجی‌ها به صورت دوقطبی.

حل: اگر هر دو ورودی یک باشد خروجی یک و در غیر این صورت منفی یک خواهد بود.

Input		Target		Weight changes			Weights		
x_1	x_2	b	y	Δw_1	Δw_2	Δb	w_1	w_2	b
--	--	--	--	--	--	--	0	0	0
1	1	1	1	1	1	1	1	1	1
1	-1	1	-1	-1	1	-1	0	2	0
-1	1	1	-1	1	-1	-1	1	1	-1
-1	-1	1	-1	1	1	-1	2	2	-2

گام‌های عملیات:

۱- تمام مقادیر وزن‌ها و bias را برابر صفر قرار می‌دهیم:

$w_1=w_2=0$, $b=0$

۲- مقادیر وزن‌ها و bios را با استفاده از رابطه مربوطه به دست می‌آوریم:

$\Delta w_i = x_i y$

تعیین ناحیه مرزی:

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

$$b + x_1 w_1 + x_2 w_2 = 0$$

$$x_2 = -x_1 \frac{w_1}{w_2} - \frac{b}{w_2}$$

ورودی اول:

$$x_2 = -x_1 \frac{1}{1} - \frac{1}{1} = -x_1 - 1$$

ورودی دوم:

$$x_2 = -x_1 \frac{0}{w_2} = 0$$

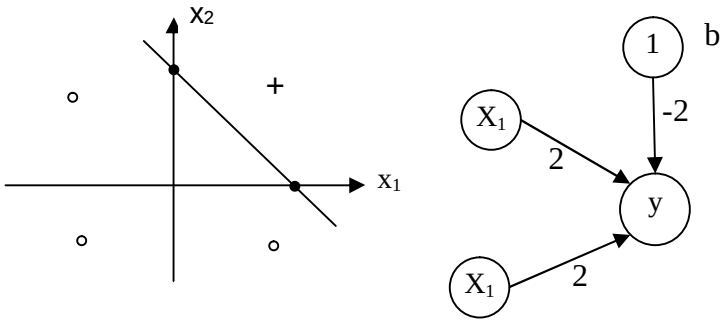
ورودی سوم:

$$x_2 = -x_1 \frac{1}{1} - \frac{-1}{1} = -x_1 + 1$$

ورودی چهارم:

$$x_2 = -x_1 \frac{2}{2} + 1 = -x_1 + 1$$

پس خط $x_2 = -x_1 + 1$ یک خط جداکننده و به عبارتی جداپذیر خطی می‌باشد.



تمرین: با استفاده از شبکه Heb دو الگوی کاراکترهای I و J را مطابق شکل زیر طبقه‌بندی کنید:

+	+	+
	+	
+	+	

J = -1

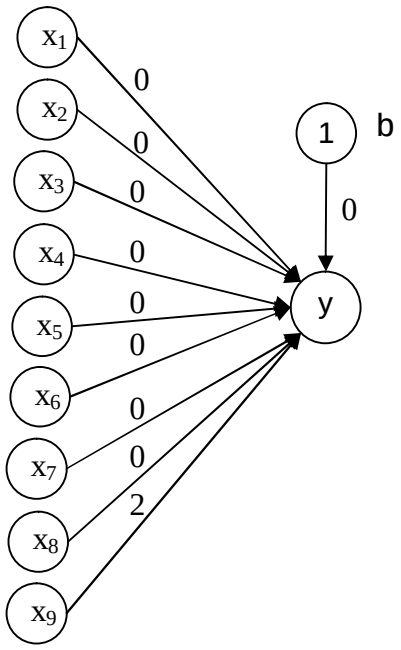
I = 1

+	+	+
	+	
+	+	+

pattern	Inputs									targets		Weight changes										Weights									
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	b	y	Δ_1	Δ_2	Δ_3	Δ_4	Δ_5	Δ_6	Δ_7	Δ_8	Δ_9	Δ	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	b
																						0	0	0	0	0	0	0	0	0	0
I	1	1	1	-1	1	-1	1	1	1	1	1	1	1	1	-1	1	-1	1	1	1	1	1	1	1	-1	1	-1	1	1	1	1
J	1	1	1	-1	1	-1	1	1	-1	1	-1	-1	-1	-1	1	-1	1	-1	-1	1	-1	0	0	0	0	0	0	0	0	2	0

اختلاف I , J در w_9 است.

شبکه Heb ی که این الگو را تشکیل می‌دهد به صورت زیر است:



تمرین: تعریف یک شبکه Heb برای تابع XOR با ورودی‌ها و خروجی‌های قطبی.

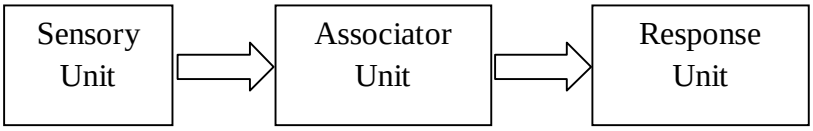
تمرین: با استفاده از شبکه Heb دو الگوی کاراکترهای E و F را مطابق شکل طبقه‌بندی کنید:

+	+	+
+		
+	+	+
+		
+	+	+

+	+	+
+		
+	+	+
+		
+		

شبکه‌های عصبی Percrptron

این شبکه توسط Rossent Blatt ، Minsky و Papert معرفی شد. ساختار یک شبکه Perceptron به صورت زیر است:



واحد Sensory و Associator دارای فعال‌ساز باینری (Binary) و واحد Response دارای فعال‌ساز bipolar, Binary سه سطحی (+1 ,0 ,-1) می‌باشد که اینجا وزن‌ها بین Associator و Response مورد بررسی قرار می‌گیرد.

توجه: آموزش و یادگیری Perceptron تا زمانی ادامه می‌یابد که خطایی وجود نداشته باشد.

مراحل الگوریتم Perceptron

گام اول: مقداردهی اولیه وزن‌ها و Bios (معمولا صفر می‌باشد) ولی می‌توان بر اساس مدل‌های فازی و الگوریتم ژنتیک (GA) نیز این مقادیر را به دست آورد و تنظیم پارامتر نرخ یادگیری (α) یا Learning rate و تعیین مقدار Threshold (θ)

توجه: α معمولا عددی بین صفر تا یک خواهد بود.

گام دوم: برای هر جفت الگوی آزمایش (S:t) گام‌های سوم تا ششم را انجام دهید.

گام سوم: تعیین مقادیر فعال‌ساز واحد ورودی ($x_i = S_i \ , 1 \leq i \leq n$)

گام چهارم: محاسبه پاسخ واحد خروجی

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

$$y = f(y_{in}) = \begin{cases} 1 & if \quad y_{in} > q \\ 0 & if \quad -q \le y_{in} \le q \\ -1 & if \quad y_{in} < -q \end{cases}$$

گام پنجم: Update وزن‌ها و bios در صورتی که مقدار هدف target (t) مساوی با مقدار خروجی (output) نباشد.

** اگر $t \neq y$ و مقدار $x_i \neq 0$ آن‌گاه:

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

و مقدار b(new)

$$b(new) = b(old) + \alpha.t$$

*** در غیر این صورت (t = y):

$$w_i(new) = w_i(old) \quad ^{o} \quad \Delta w = 0$$

$$b(new) = b(old) \; ^{o} \; \Delta b = 0$$

گام ششم: بررسی شرط خاتمه.

مثال: تعریف یک شبکه Perceptron برای تابع AND

گام اول:

$$w_1=w_2=b=0 \ , \ \alpha = 1 \ , \ \theta = 0$$

گام دوم:

$$(1:1):1$$

گام سوم:

$$x_i = (1,1)$$

گام چهارم:

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

$Y_{in} = 0$ در نتیجه:

$$y = f(y_{in}) = \begin{cases} 1 & if \quad y_{in} > 0 \\ 0 & if \quad -0 \le y_{in} \le 0 \\ -1 & if \quad y_{in} < -0 \end{cases}$$

گام پنجم: چون $t \neq y$ $(t = 1 \ , \ y = 0)$ در نتیجه داریم:

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

$w_1(new)= w_1(old) + \alpha.t.x_1= 0 + 1\times1\times1 = 1$

$w_2(new)= w_2(old) + \alpha.t.x_2= 0 + 1\times1\times1 = 1$

$b(new) = b(old) + \alpha.t = 0 + 1 = 1$

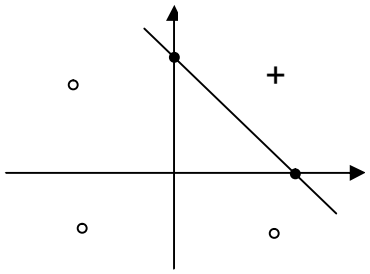
به همین ترتیب برای سایر ورودی‌ها عمل می‌کنیم:

Inputs		Target				Weight changes			Weights		
x_1	x_2	b	t	y_{in}	y	Δw_1	Δw_2	Δb	w_1	w_2	b
--	--	--	--	--	--	--	--	--	0	0	0
1	1	1	1	0	0	1	1	1	1	1	1
1	-1	1	-1	1	1	-1	1	-1	0	2	0
-1	1	1	-1	2	1	1	-1	-1	1	1	-1
-1	-1	1	-1	-3	-1	0	0	0	1	1	-1

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

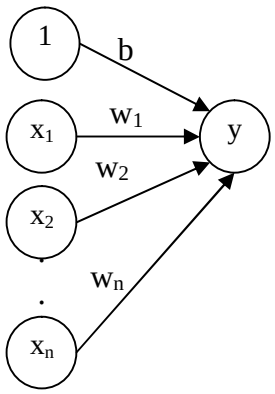
$$b + x_1 w_1 + x_2 w_2 = 0$$

$$x_2 = -x_1 \frac{w_1}{w_2} - \frac{b}{w_2} \quad \Rightarrow \quad x_2 = -x_1 + 1$$



شبکه‌های عصبی Adaline

این شبکه توسط Windrow و Hoff ارایه شده است (۱۹۶۰) که از مقادیر bipolar در ورودی و خروجی آن استفاده می‌شود. مقادیر وزن این شبکه قابل تنظیم و قوانین یادگیری مشهوری که برای این شبکه به کار می‌روند عبارتند از قانون Delta rule ، LMS و قانون Windrow , Hoff ساختار این شبکه به صورت زیر است:



توجه: مقادیر وزن این شبکه Random و غیر صفر می‌باشد. که معمولا با اعداد کوچک وزن‌دهی می‌شوند.

قانون دلتا (Delta rule)

بر اساس این قانون وزن شبکه ، آن‌قدر تغییر می‌کند تا تفاوت بین ورودی (x) و خروجی شبکه (y) با تابع هدف (t) حداقل (minimum) شود و داریم:

$$\Delta w_i = \alpha.(t - y_{in}).x_i$$

و همچنین مقدار خطای خطای LMS (Least Mean Square) از رابطه زیر محاسبه می‌شود:

$$E = \sum_j (t_j - y_j)^2$$

مراحل الگوریتم Adaline

۱- مقداردهی اولیه تمام وزن‌ها (غیر صفر) اما مقادیر کوچک تصادفی

۲- اگر شرط خاتمه درست نیست گام‌های سه به بعد را انجام دهید:

۳- برای هر جفت ورودی و هدف (S:t) گام‌های چهارم تا ششم را انجام دهید.

۴- مقدار فعال‌ساز برای ورودی‌های

$$x_i = s_i \quad (1 \le i \le n)$$

۵- محاسبه رابطه ورودی و خروجی هر واحد شبکه با استفاده از رابطه زیر:

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

۶- تنظیم مجدد وزن‌ها با استفاده از روابط ذیل مبتنی بر قانون دلتا:

$$w_i(new) = w_i(old) + \alpha.(t - y_{in}).x$$

$$b(new) = b(old) + \alpha.(t - y_{in})$$

۷- بررسی شرط خاتمه

توجه: برای محاسبه خروجی شبکه از تابع فعال‌ساز ذیل استفاده می‌شود:

$$y = f(y_{in}) = \begin{cases} +1 & if \quad y_{in} \ge 0 \\ -1 & if \quad y_{in} < 0 \end{cases}$$

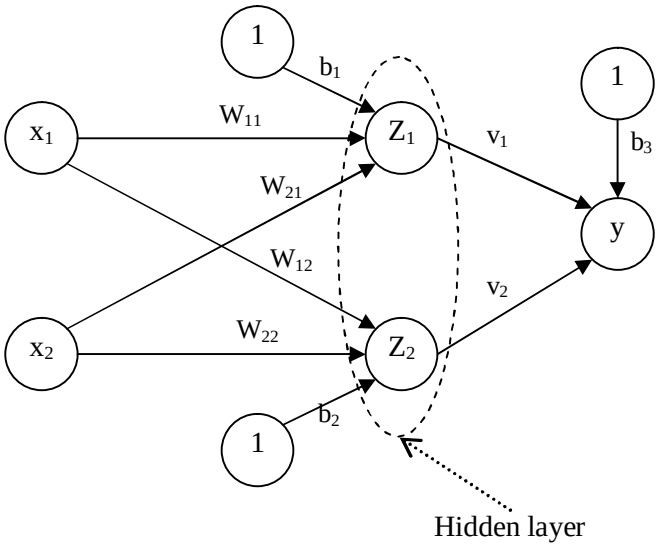
مثال: استفاده از شبکه‌های Adaline برای الگوهای ورودی و هدف مطابق جدول ذیل:

	Inputs			**Target**	**Net**		**Weights changes**			**weights**			**Error (E)**
	x_1	x_2	**b**	**t**	y_{in}	t_i-y_i	Δw_1	Δw_2	Δb	w_1	w_2	**b**	$(t_i-y_i)^2$
Epoch1= 5.71	1	1	1	-1	0.6	-1.6	-0.32	-0.32	-0.32	-0.12	-0.12	-0.12	2.56
	1	-1	1	1	-0.12	1.12	0.22	-0.22	-0.22	0.1	-0.34	0.1	1.25
	-1	1	1	-1	-0.34	-0.66	0.13	-0.13	-0.13	0.24	-0.48	-0.03	0.43
	-1	-1	1	-1	0.21	-1.2	0.24	-0.24	-0.24	0.48	-0.23	-0.27	1.47

	Inputs			Target	Net		Weights changes			weights			Error (E)
	x_1	x_2	b	t	y_{in}	t_i-y_i	Δw_1	Δw_2	Δb	w_1	w_2	b	$(t_i-y_i)^2$
Epoch2=													

شبکه‌های عصبی Madaline

این شبکه ترکیبی از شبکه‌های عصبی Adaline می‌باشد که به نام شبکه Multi Layer Adaline یا آدالاین چندلایه نیز مشهور است. ساختار این شبکه به صورت زیر است:



مطابق شکل فوق این شبکه از دو ورودی (x_1,x_2) و دو واحد مخفی hidden (Z_1,Z_2) و یک خروجی تشکیل شده است.

الگوریتم MR1 برای شبکه‌های Madaline:

نکات:

۱- در این الگوریتم مقادیر وزن خروجی شبکه ثابت (V) ولی مقادیر وزن لایه مخفی (w) تغییر می‌کند.

۲- مقادیر وزن‌های v_1 و v_2 خروجی (y) و مقدار bios ، b_3 با مقدار ثابت 0.5 مقداردهی می‌شود.

۳- تابع فعال ساز (Activation Function) برای Z_1 ، Z_2 و y برابر است با:

$$f(p)=\begin{cases}1 & if \quad p\geq 0\\ 0 & if \quad p<0\end{cases}$$

گام‌های الگوریتم

۱- مقداردهی اولیه وزن‌ها ، bios و α (0.5) برای $b_3= 0.5$ $v_1 = v_2$ و برای سایر وزن‌ها (w) مقادیر تصادفی کوچکی انتخاب می‌شود. (مقادیر random نزدیک به صفر)

۲- اگر شرط خاتمه نادرست بود ، گام‌های سوم تا نهم را انجام دهید.

۳- برای هرجفت ورودی دوقطبی (s:t) گام‌های چهارم تا هشتم را انجام دهید.

۴- تنظیم فعال‌ساز برای واحدهای ورودی $(x_i = s_i \ , \ 1 \le i \le n)$

۵- محاسبه ورودی به واحدهای مخفی hidden (منظور Z)

$$Z_{in1} = b_1 + x_1.w_{11} + x_2.w_{21}$$

$$Z_{in2} = b_2 + x_1.w_{12} + x_2.w_{22}$$

۶- پیدا کردن خروجی واحدهای مخفی (Z)

$$\begin{cases} Z_1 = f(Z_{in1}) \\ Z_2 = f(Z_{in2}) \end{cases}$$

۷- محاسبه ورودی به خروجی شبکه (y)

$$Y_{in} = b_3 + Z_1.v_1 + Z_2.v_2$$

و تعیین خروجی شبکه

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

۸- یافتن مقدار خطا (error) و تعیین وزن‌های جدید با توجه به شرایط زیر:

- ♣ اگر $t = y$ باشد تغییری در وزن‌ها ایجاد نمی‌شود.
- ♣ اگر $t \neq y$ باشد آن‌گاه:

- اگر $t = 1$ ، آن‌گاه تغییر وزن‌های Zj که ورودی شبکه نزدیک‌ترین مقدار به صفر را دارد از روابط زیر به دست می‌آید:

$$w_{ij}\ (new) = w_{ij}\ (old) + \alpha.(1-Z_{inj}).x_i$$

$$b_j(new) = b_j(old) + \alpha.(1-Z_{inj})$$

- اگر $t = -1$ ، آن‌گاه تغییر وزن‌های Z_k که ورودی شبکه مثبت دارند از روابط ذیل به دست می‌آید:

$$w_{ik}\ (new) = w_{ik}\ (old) + \alpha.(1-Z_{ink}).x_i$$

$$b_k(new) = b_j(old) + \alpha.(1-Z_{ink})$$

۹- بررسی شرط خاتمه (شرط خاتمه می‌تواند ثبات وزن‌ها یا تعداد Epochهای مشخص باشد).

مثال: طراحی یک شبکه MADALINE برای تابع XOR با استفاده از MR1

۱- مقداردهی اولیه

$$v_1= v_2 = b_3 = \alpha = 0.5$$

$$w_{11}= 0.05 \qquad w_{21}=0.2 \qquad b_1=0.3$$

$$w_{12}= 0.1 \qquad w_{22}= 0.2 \qquad b_2= 0.15$$

۲- آغاز آموزش

۳- برای جفت ورودی 1-:(1:1) گام‌های چهارم تا هشتم را انجام می‌دهیم:

۴- $x= (1, 1)$ و $x_i= s_i$

۵- محاسبه ورودی به واحد مخفی

$$Z_{in1}= b_1 + x_1.w_{11}+ x_2.w_{21} = 0.3 + 1\times0.05 + 1\times0.2 = 0.55$$

$$Z_{in2}= b_2 + x_1.w_{12}+ x_2.w_{22} = 0.15 + 1\times0.1 + 1\times0.2 = 0.45$$

۶- یافتن خروجی Z_1 , Z_2 با استفاده از تابع فعال‌ساز:

$$Z_1= f(Z_{in1}) = 1 \quad , \quad Z_2= f(Z_{in2}) = 1$$

۷- محاسبه ورودی به خروجی شبکه

$$y_{in}= b_3 + v_1.Z_1 + v_2.Z_2= 0.5 + 1\times0.5 + 1\times0.5 = 1.5$$

$$Y = f(y_{in}) = 1$$

۸- با توجه به این‌که $t \neq y$ $(y = 1 , t = -1)$ وزن‌های Z1 , Z2 تغییر می‌کند تا ورودی شبکه مثبت شود.

$$w_{11}(new) = w_{11}(old) + \alpha.(-1 - Z_{in1}) \times1 = 0.05 + 0.5(-1-0.55)\times1= 0.725$$

$$w_{12}(new) = w_{12}(old) + \alpha.(-1 - Z_{in2}) \times1= 0.1 + 0.5(-1-0.45)\times1= 0.625$$

$$b_1(new) = -0.475 \qquad w_{22}(new) = -0.522$$

$$w_{21}(new) = -0.575 \qquad b_2(new) = -0.575$$

۹- بررسی شرط خاتمه که در این مسأله پس از ۳ مرحله (Epoch) مقادیر وزن‌ها پایدار می‌شود.

	Inputs		target													
	x_1	x_2	b	t	Z_{in1}	Z_{in2}	w_{11}	w_{21}	b_1	w_{12}	w_{22}	b_2	Z_1	Z_2	y_{in}	y
Epoch1	1	1	1	-1	0.55	0.45	-0.725	-0.575	-0.475	-0.625	-0.525	-0.575	1	1	1.5	1
	1	-1	1	1	-0.625	-0.675	0.0875	-1.387	0.337	-0.625	-0.525	-0.575	-1	-1	-0.5	-1
	-1	1	1	-1	-1.1375	-0.47	0.087	-1.38	0.337	-1.362	0.212	0.162	-1	-1	-0.5	-1
	-1	-1	1	-1	1.637	1.31	1.46	-0.068	0.98	-0.207	1.36	-0.994	1	1	1.5	1