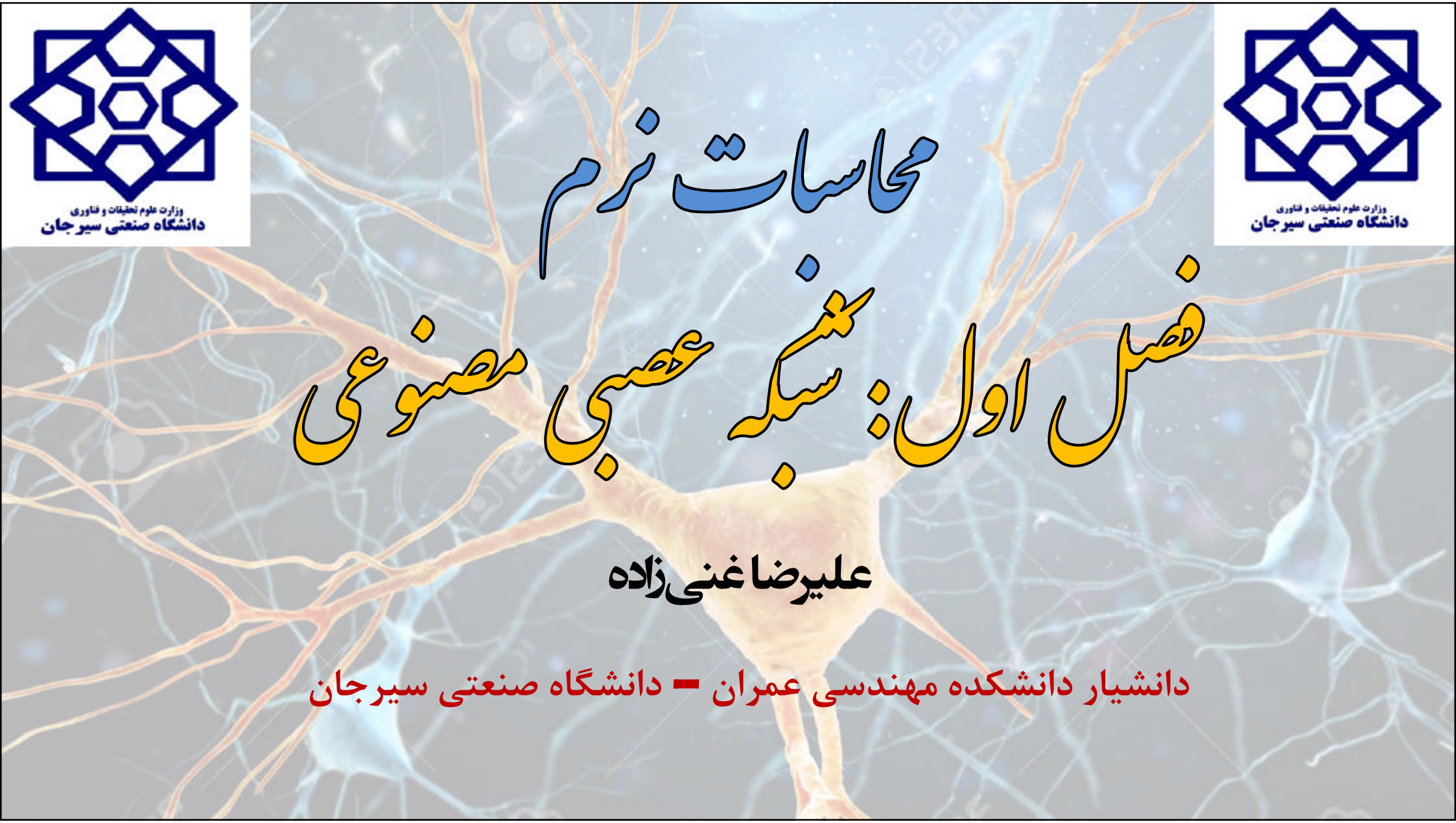


محاسبات نرم

فصل اول: شبکه عصبی مصنوعی

علیرضا غنی‌زاده

دانشیار دانشکده مهندسی عمران – دانشگاه صنعتی سیرجان

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir

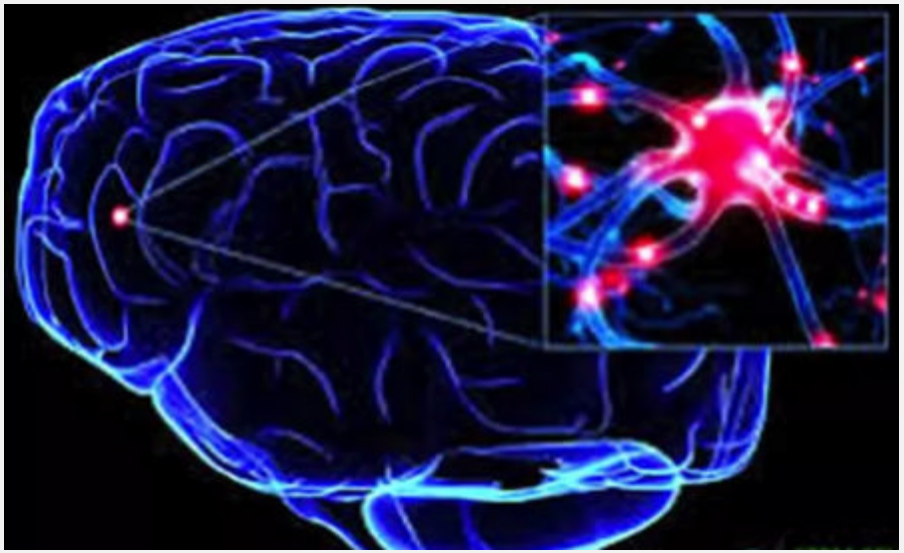
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مقدمه

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

سیستم اعصاب مرکزی و مغز

- **مغز یک سیستم پردازشگر اطلاعات موازی، غیر خطی و بسیار پیچیده است.**



کارهایی که مغز انسان انجام می دهد

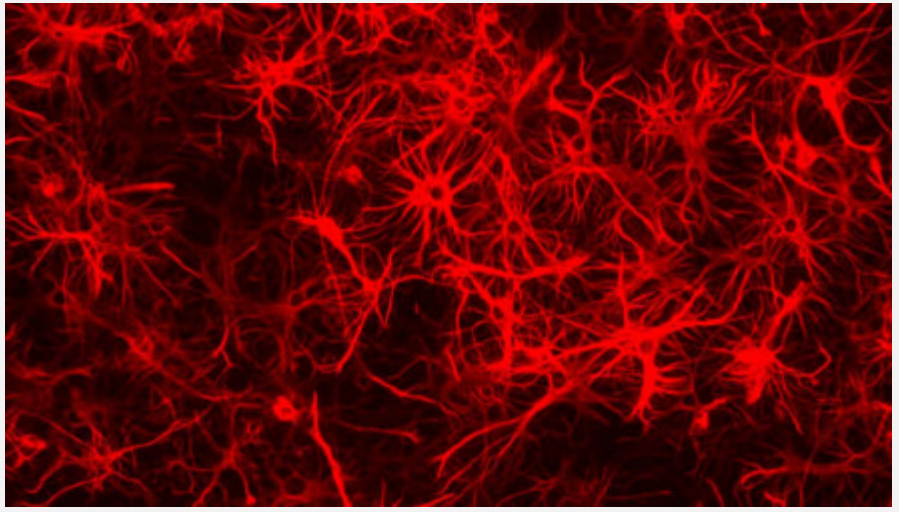
- یادگیری (تشخیص چهره)
- ذخیره سازی اطلاعات
- تصمیم گیری
- پیش بینی
- محاسبه
- . . .

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

سیستم اعصاب مرکزی و مغز

مغز = شبکه ای بسیار بزرگ از عصب‌ها (نرون‌ها)

- ۱۰ میلیارد نرون
- ۶۰ تریلیون اتصال بینابینی
- شبکه عصبی مصنوعی = شبیه سازی شبکه عصبی طبیعی



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

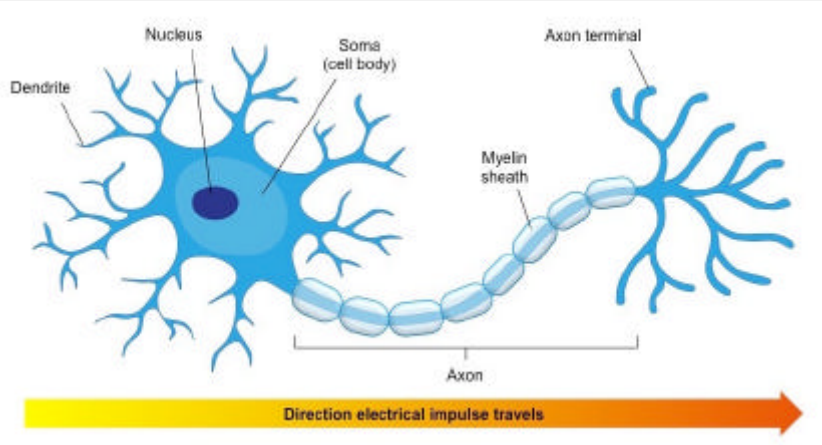
نرون طبیعی

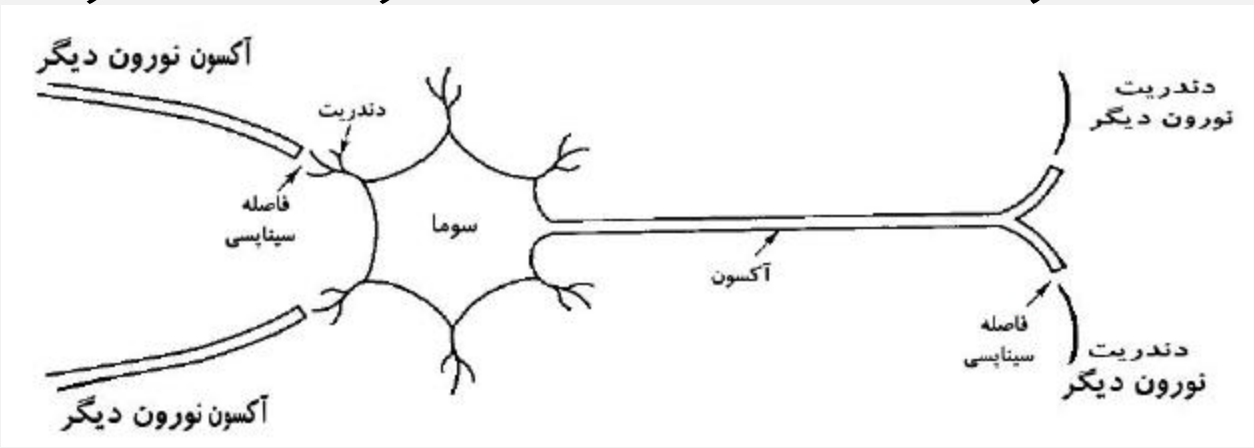
عنصر پردازشگر مغز:

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

سه جزء تشکیل دهنده یک نرون طبیعی

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





Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

عملکرد نرون طبیعی

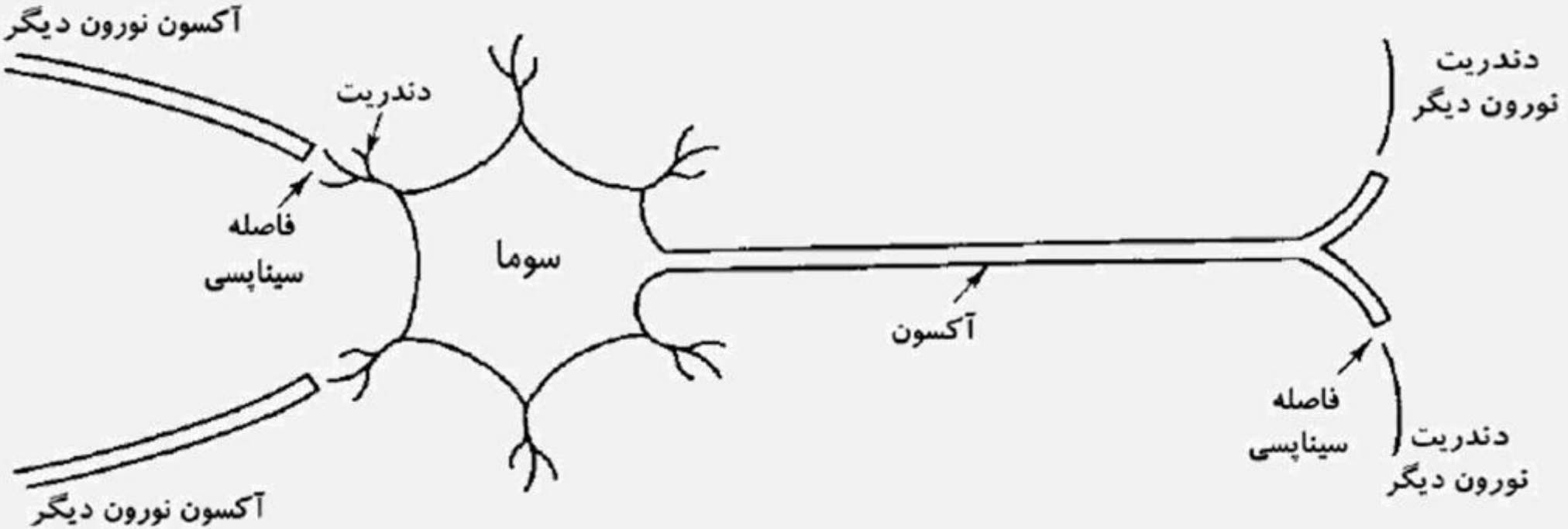
- دریافت سیگنال از سایر نرون ها توسط دندریت‌ها
- عبور سیگنال‌ها با یک فرآیند شیمیایی از فاصلهٔ سیناپسی (Synaptic Gap)
- عمل شیمیایی انتقال دهنده، سیگنال ورودی را تغییر می‌دهد (تضعیف/تقویت سیگنال)
- سوما سیگنال های ورودی به سلول را جمع می‌بندد
- زمانی که یک سلول به اندازه کافی ورودی دریافت نماید، برانگیخته می‌شود و سیگنالی را
- از آکسون خود به سلول های دیگر می‌فرستد.
- انتقال سیگنال از یک نرون خاص نتیجهٔ غلظت‌های مختلف یون‌ها در اطراف پوشش آکسون نرون (ماده سفید) مغز می باشد.

یون ها = پتاسیم، سدیم و کلرید

- سیگنال ها به صورت ضربه‌های الکتریکی هستند

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

عملکرد نرون طبیعی



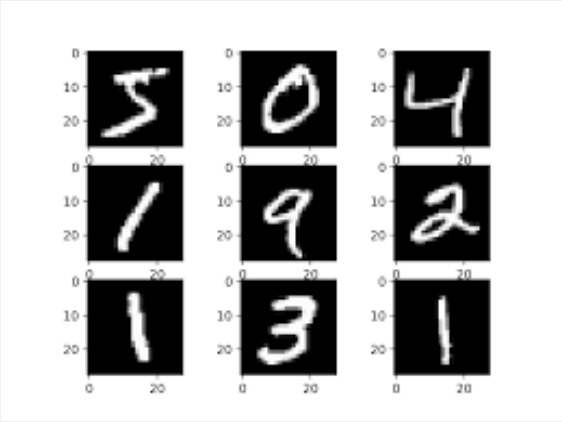
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

خلاصه ویژگی‌ها و خصوصیات نرون طبیعی

- جزء پردازشگر (نرون) سیگنال های 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 نرون های دیگر (شاخه‌های آکسون) برود.
- پردازش اطلاعات به صورت محلی صورت می‌گیرد.
- مفهوم حافظه در اجزای مختلف نرون توزیع می‌شود:
- حافظهٔ بلند مدت در سیناپس‌ها یا وزن های نرون قرار می‌گیرد.
- حافظهٔ کوتاه مدت با سیگنال های فرستاده شده توسط نرون ها مطابقت دارد.
- توانایی سیناپس می‌تواند با آزمایش و کسب تجربه تغییر کند.
- انتقال دهنده‌های عصبی برای سیناپس ها می توانند تحریک کننده (Excitatory) یا بازدارنده (Inhibitory) باشند.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

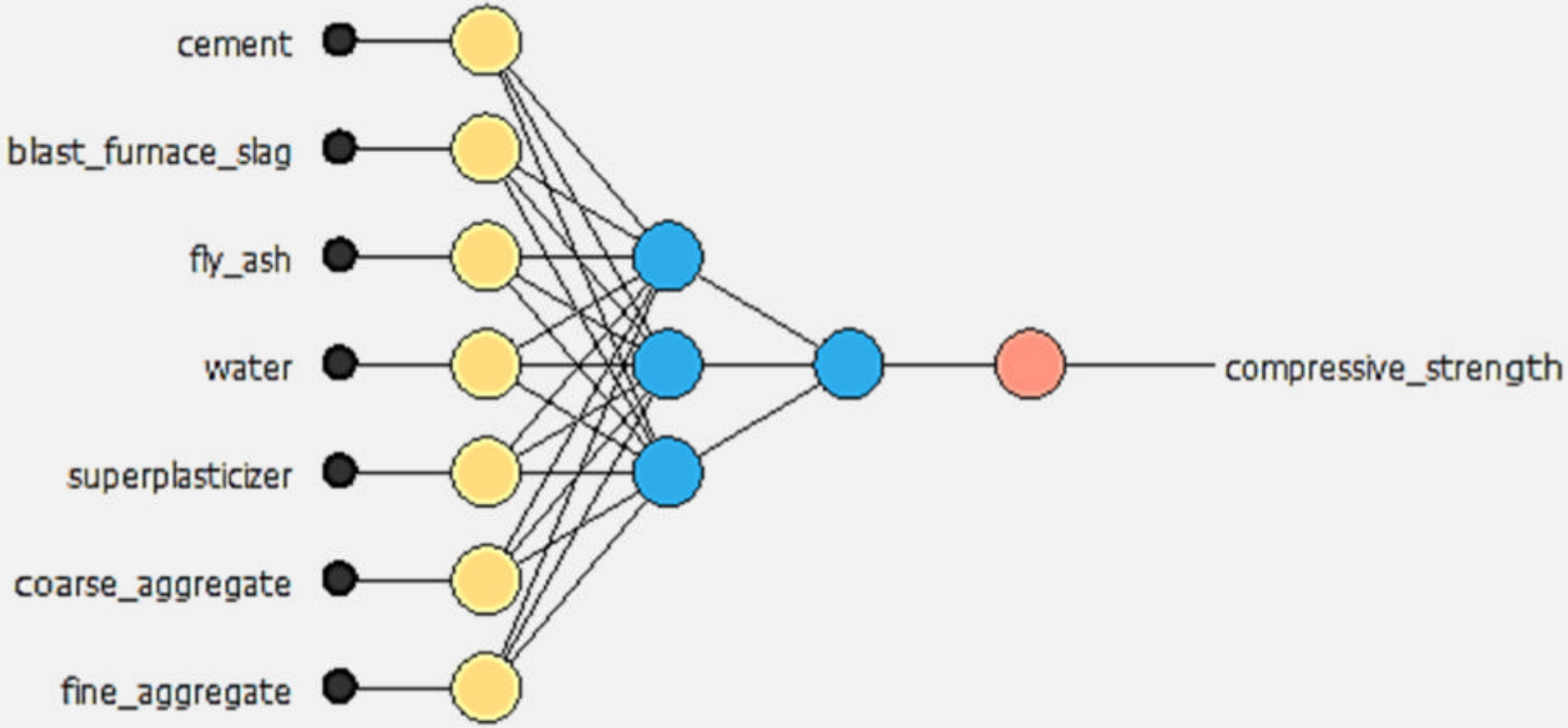
معرفی شبکه عصبی مصنوعی (Artificial Neural Network)



- شبکهٔ عصبی مصنوعی در واقع سیستمی الهام گرفته شده از مغز انسان است.
- مدل ریاضی تعمیم یافتهٔ تشخیص انسان بر اساس زیست شناسی عصبی.
- شبکه عصبی مصنوعی همانند مغز انسان توانایی آموزش و یادگیری دارد.
- مهمترین کاربردهای شبکه عصبی شامل پیش‌بینی (رگرسیون و طبقه بندی) و شناسایی الگو است
- شبکه‌های عصبی مصنوعی برای مدلسازی رابطه بین پارامترهای ورودی و پارامترهای خروجی که تعریف رابطه بین آن‌ها از دیدگاه مهندسی بسیار پیچیده و غیر خطی است بسیار کارا می‌باشند.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

نمونه‌ای از کاربرد شبکه عصبی در پیش‌بینی مقاومت فشاری بتن



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

- دهه ۴۰ – اولین شبکه‌های عصبی مصنوعی
  - ۱۹۴۳– معرفی نرون مک‌کلاچ–پیتز (اولین شبکه عصبی مصنوعی)
    - توسط وارن مک‌کلاچ 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

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

- دهه ۶۰ – گسترش پرسپترون + آدالاین
  - ۱۹۶۰ – شبکه آدالاین
    - آدالاین (ADALINE) - نرون خطی وفقی (ADAptive LInear NEuron) یا سیستم خطی وفقی (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 ، وزن‌هاي اتصال آن واحد تنظیم مي‌شود ، اما قانون دلتا وزن‌ها را طوری تنظیم مي‌کند تا اختلاف بین خروجی شبکه و خروجي مطلوب کمینه کند
    - مادالاین: شکل توسعه یافته و چندلایهٔ 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

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

- دهۀ 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 حافظۀ پیوندی
- ۱۹۷۷- تحقیقات آندرسن از دانشگاه براون در زمینۀ شبکه‌هایی عصبی حافظۀ انجمنی و انتشار نظریاتش با نام «حالت مغز در یک جعبه» (Brain-State-in-a-Box)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه 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 برندهٔ جایزهٔ نوبل در رشتهٔ فیزیک و عضو مؤسسهٔ فن‌آوری کالیفرنیا
      - به همراه دیوید تانك ، محقق AT&T
    - شبکهٔ عصبی با وزن‌های ثبات و فعال‌سازی وفقی (جزو شبکه‌های حافظهٔ انجمنی)
    - حل مسائل ارضای محدودیت مانند «مسئلهٔ فروشندهٔ دوره‌گرد»
  - نگاشت‌های خودسازمانده کوهونن (SOM)
    - توسط کوهونن از دانشگاه هلسینکی
    - استفاده در بازشناسی گفتار کلمات فنلاندی و ژاپنی ، حل «مسئلهٔ فروشندهٔ دوره‌گرد» و آهنگ‌سازی

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

- دهه ۸۰- شکوفایی شبکه‌های عصبی . . .
  - شبکه‌های نظریهٔ نوسان وفقی (ART)
    - توسط کارپنز و با همکاری گراس‌برگ
    - 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)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

- دهه ۸۰- شکوفایی شبکه‌های عصبی . . .
  - مطالعات ریاضیاتی و زیست‌شناختی
    - گراس‌برگ (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

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

تاریخچه شبکه عصبی مصنوعی

- دهه ۹۰ – دهه کاربرد
  - به کار گیری شبکه‌های عصبی در کاربردهای مختلف
  - توسعه شبکه توابع پایه شعاعی (RBF)
  - ماشین بردار پشتیبان (SVM)
- ۲۰۰۰ به بعد
  - یادگیری عمیق (Deep Learning)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه عصبي مصنوعي و مدل پرسپترون

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

ویژگی‌های شبکه عصبی مصنوعی

قابلیت یادگیری

- امکان فراگیری الگوی نگاشت از یک مجموعه ورودی به یک مجموعه خروجی
- یادگیری مسائل پیچیده و غیر خطی از طریق اجتماع نورون‌ها
- منظور از یادگیری تنظیم پارامترهای شبکه عصبی (وزن‌های اتصالی و بایاس اتنرورون‌ها) بر اساس اطلاعات موجود و استفاده از شبکه آموزش دیده جهت پیش‌بینی سایر موارد جدید است

قابلیت تعمیم (Generalization)

- پس از آموزش شبکه عصبی بر اساس مثال‌های داده شده، شبکه تابع نگاشت بین ورودی‌ها و خروجی را فرا می‌گیرد و می‌تواند با معرفی یک ورودی جدید خروجی مناسب را تولید کند.

مقاوم بودن در برابر خطا (Fault Tolerance)

- پردازش اطلاعات در نورون‌های مستقل سبب می‌شود تا خطای محلی بر روی خروجی نهایی تاثیر قابل ملاحظه‌ای نداشته باشد. به عبارت دیگر نورون‌ها خطاهای محلی یکدیگر را اصلاح می‌کنند.

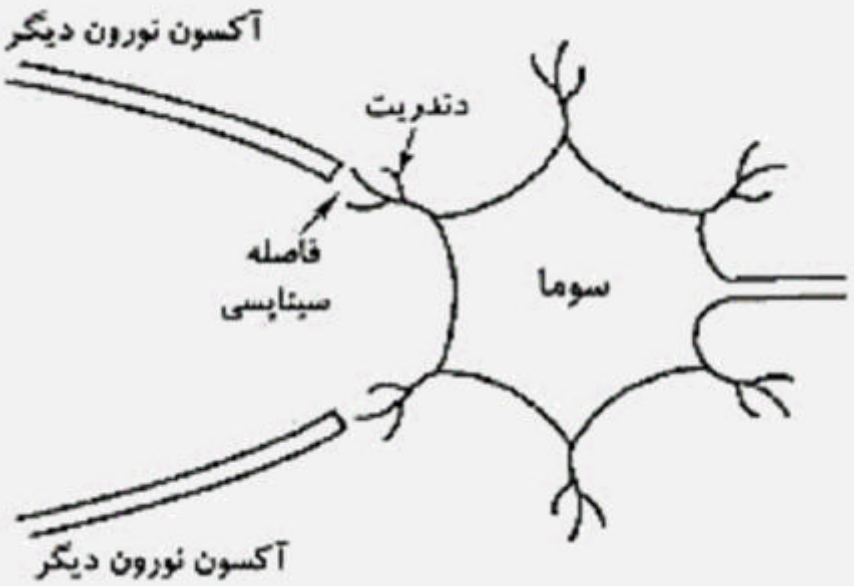
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

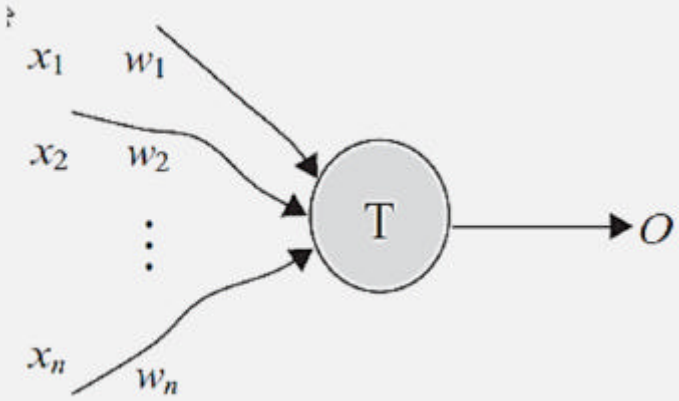
فرضیات پایه شبکهٔ عصبی مصنوعی

- 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

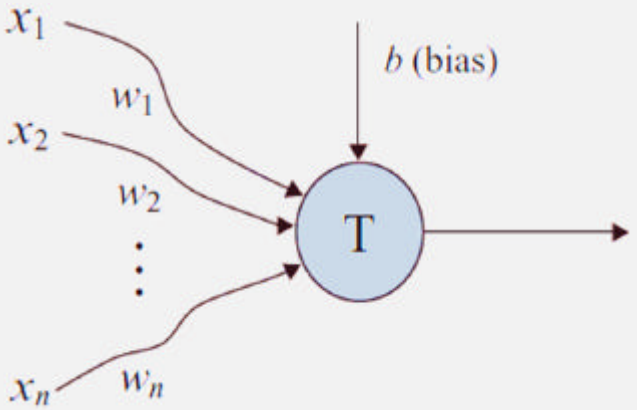
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

نرون مصنوعی





McCulloch and Pitts' neuron model

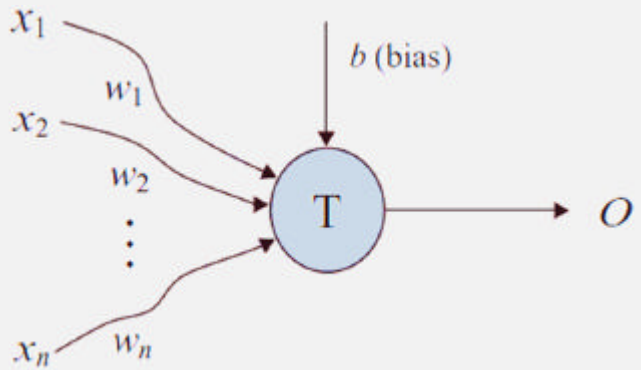


$$O = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} w_i x_i - b \geq T \\ 0 & \text{if } \sum_{i=1}^{n} w_i x_i - b < T \end{cases}$$

Rosenblatt's perceptron model

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

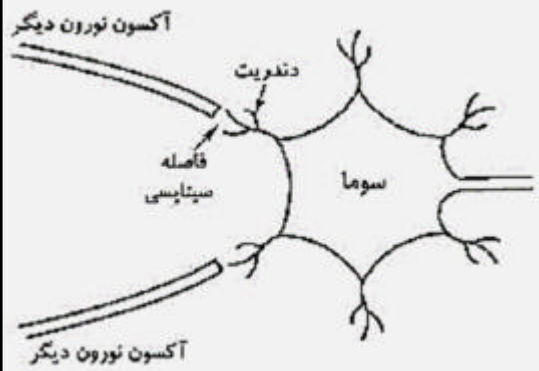
مدل پرسپترون (Perceptron)

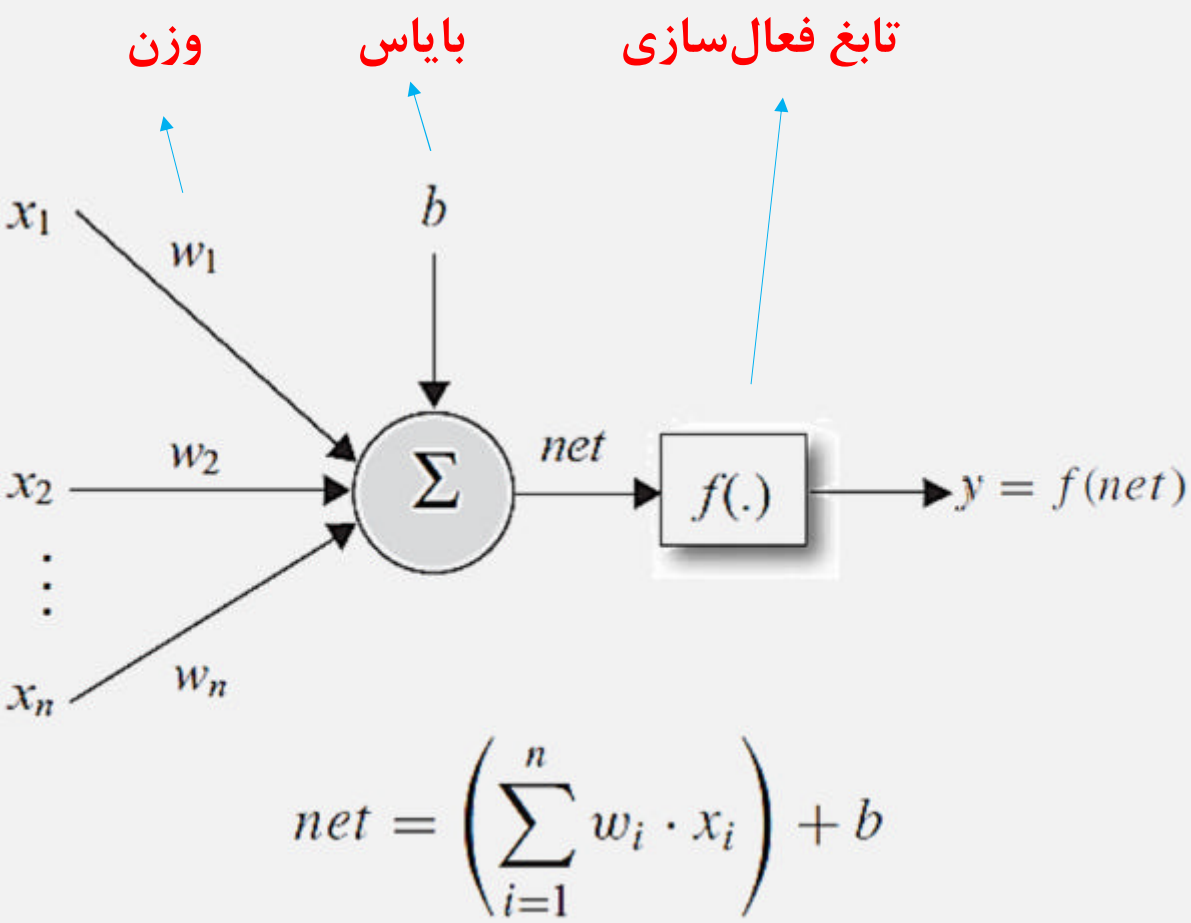


شبکه عصبی طبیعی	شبکه عصبی مصنوعی
سوما	نورون
دندریت	ورودی یا اتصال بین نرون ها
آکسون	خروجی
سیناپس	وزن
جمع بستن سیگنال‌های ورودی در سوما	جمع وزن دار سیگنال‌های ورودی و بایاس در نورون
برانگیخته شدن نورون و ارسال سیگنال از آکسون	تابع فعال‌سازی (انتقال)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مدل پرسپترون (Perceptron)





Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

توابع فعال‌سازی (Activation Functions)

Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$		satlin

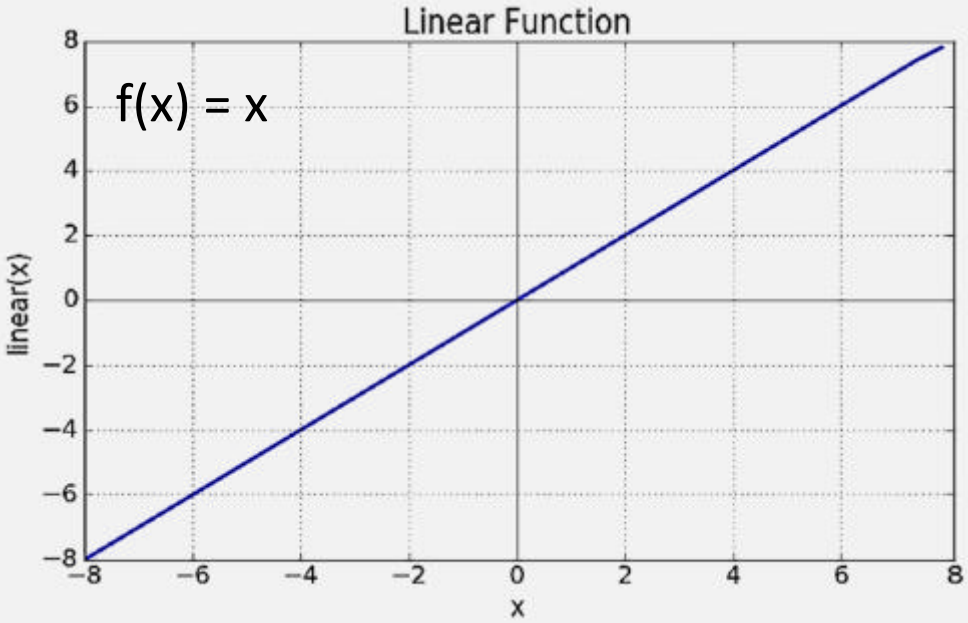
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

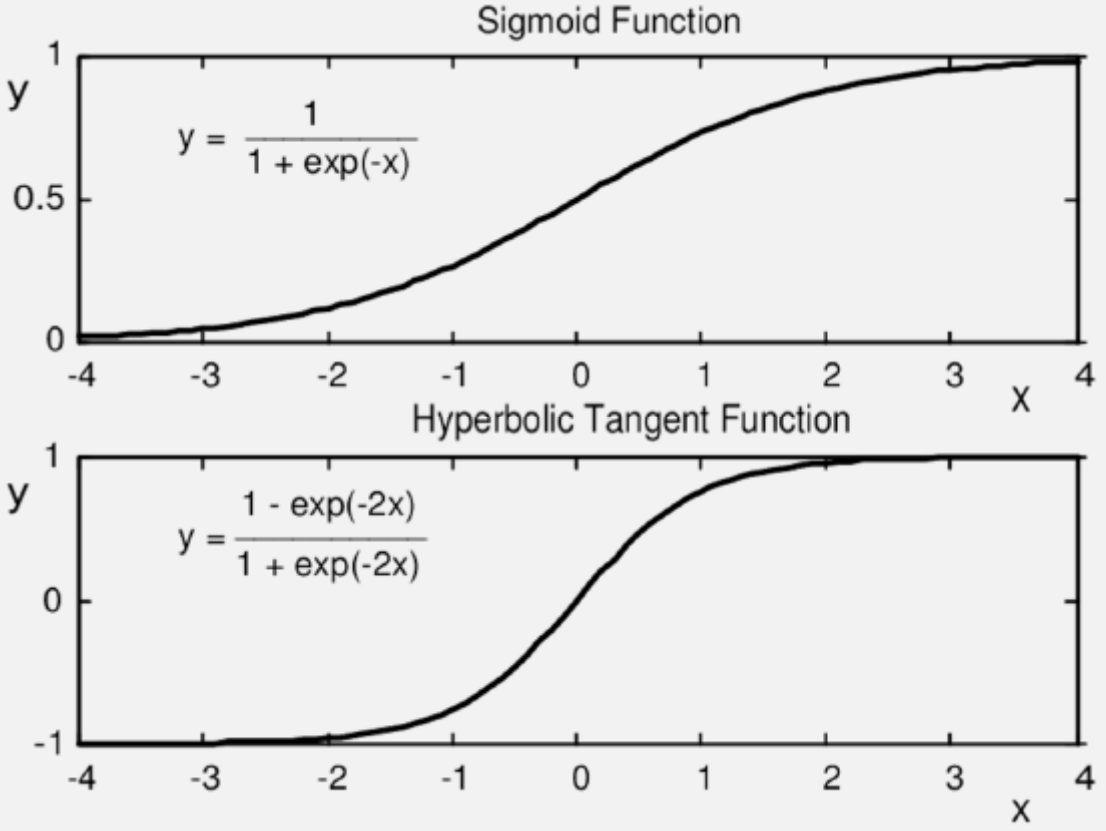
توابع فعال‌سازی (Activation Functions)

Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \le n \le 1$ $a = 1 \quad n > 1$		satlins
Log-Sigmoid	$a = \frac{1}{1+e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^{n}-e^{-n}}{e^{n}+e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \le n$		poslin

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

توابع فعال‌سازی (Activation Functions)





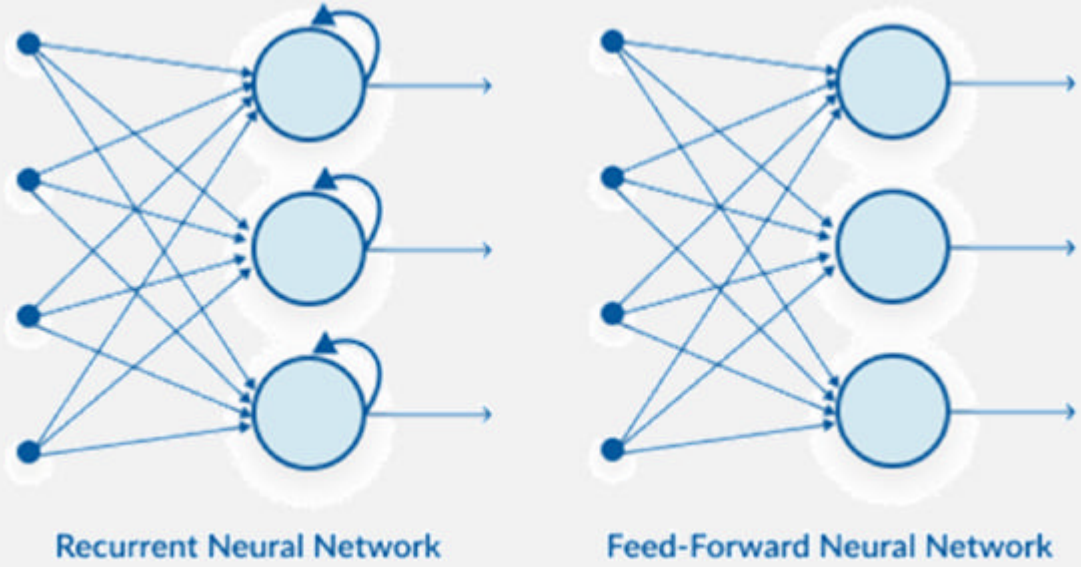
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

انواع معماري شبكه هاي عصبي مصنوعي

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

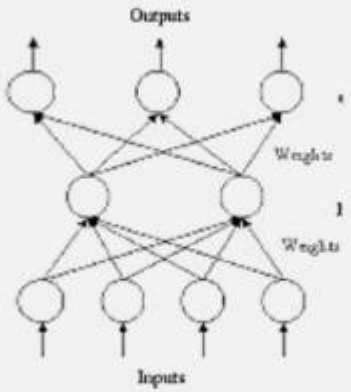
انواع ساختار یا معماری شبکه عصبی مصنوعی

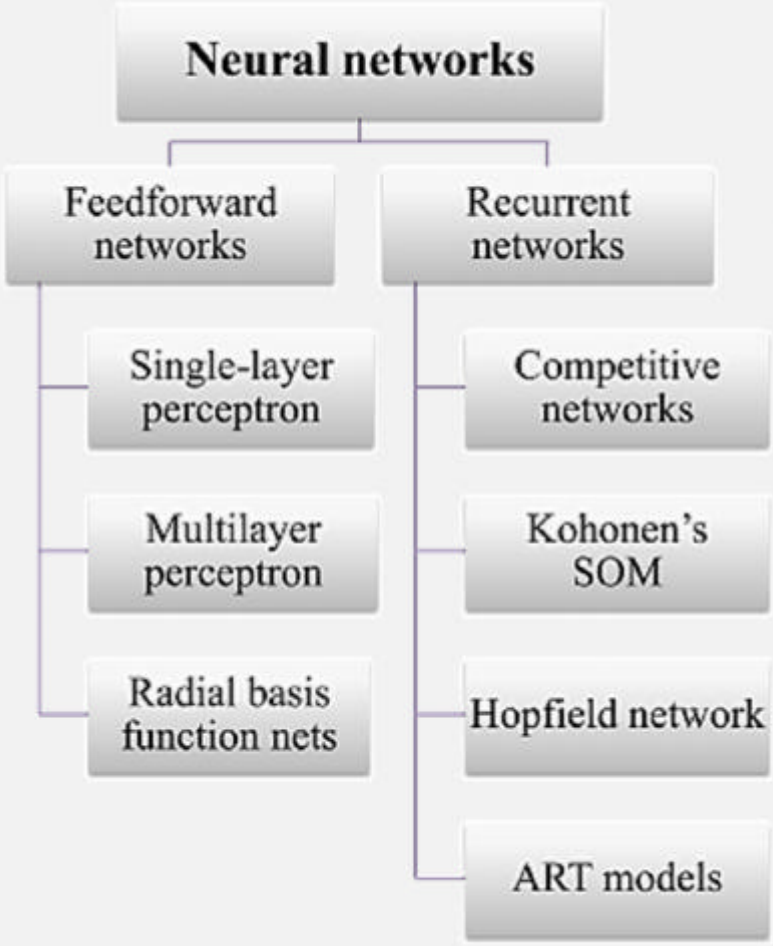
- ساختار یا معماری: آرایش و تعداد نرون‌ها در هر لایه و الگوی ارتباط داخلی بین لایه‌ها
- شبکه پیش خور (Feedforward): سیگنال‌ها در یک جهت و از سمت نرون‌های ورودی به سمت نورون خروجی (به سمت جلو) می‌روند.
- شبکه بازگشتی (Recurrent): مسیر بسته سیگنال از یک نورون به خودش یا به سایر نورون ها وجود دارد.

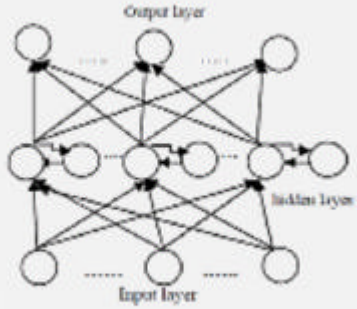


Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مهمترین معماری‌های شبکه‌های عصبی مصنوعی برای کاربرد در مهندسی عمران







Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

انواع شبکه‌های عصبی پیش‌خور

- Multilayer perceptron networks,
- Radial basis function networks,
- Generalized regression neural networks,
- Probabilistic neural networks,
- Belief networks,
- Hamming networks and
- Stochastic networks.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

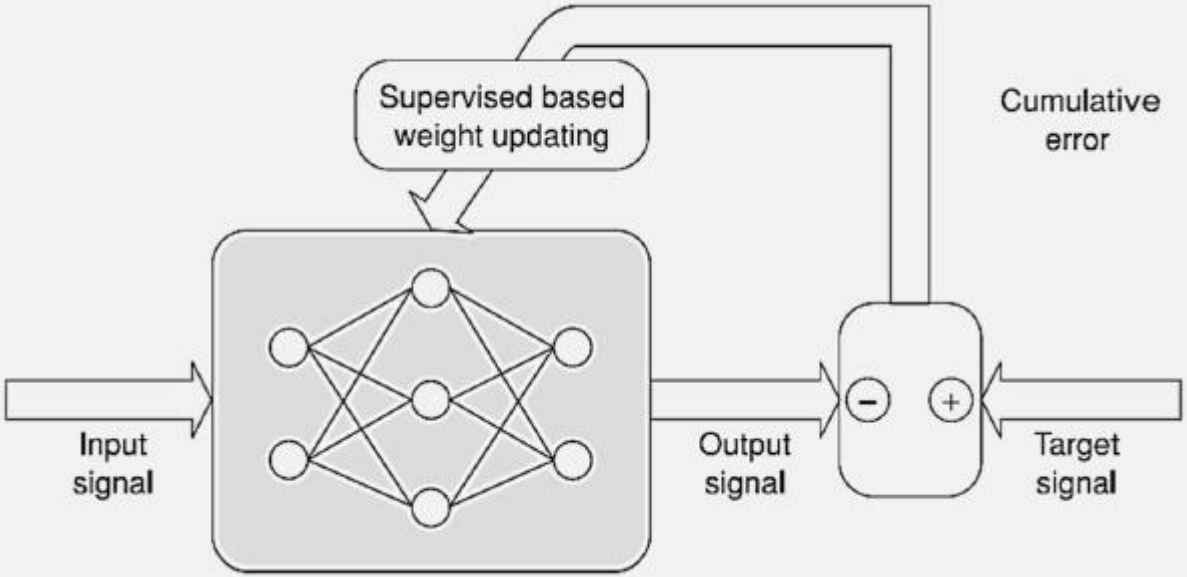
انواع الگوریتم های آموزش شبکه‌های عصبی مصنوعی

- آموزش نظارت شده (Supervised Learning)
- آموزش نظارت نشده (Unsupervised Learning)
- یادگیری تقویت شده (Reinforced Learning)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش نظارت شده

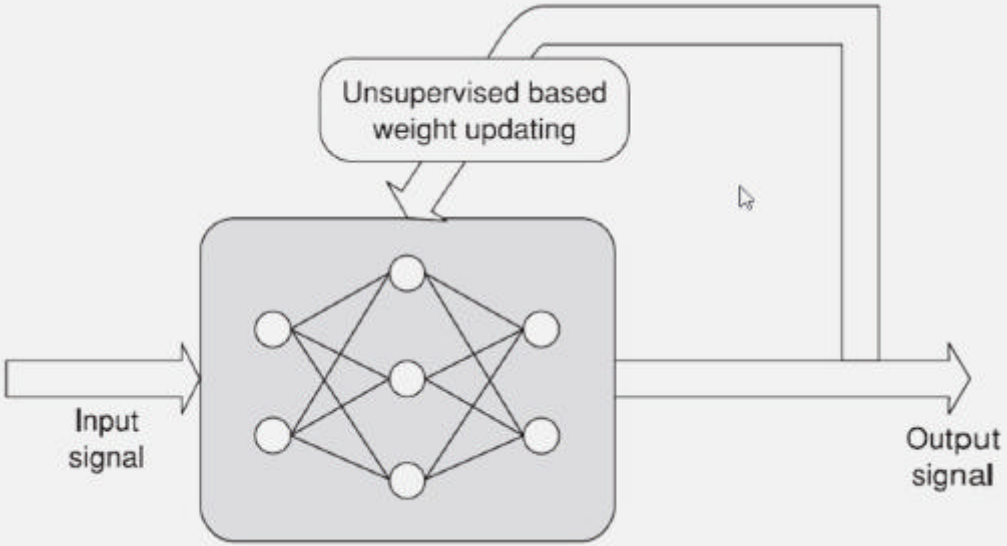
- این روش، یک روش عمومی در یادگیری ماشین است که در آن به یک سیستم، مجموعه ای از جفت‌های ورودی – خروجی ارائه شده و سیستم تلاش می‌کند تا تابعی از ورودی به خروجی را فرا گیرد.
- یادگیری تحت نظارت نیازمند تعدادی داده ورودی به منظور آموزش سیستم است.



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش نظارت نشده

- «یادگیری نظارت نشده» (Unsupervised Learning) یک دسته از روش‌های «یادگیری ماشین» (Machine Learning)برای کشف الگوهای موجود در میان داده‌ها است. داده‌های ارائه شده به الگوریتم نظارت نشده دارای برچسب نیستند، بدین معنا که متغیر ورودی X بدون هیچ متغیر خروجی متناظری داده شده است. در یادگیری نظارت شده، الگوریتم‌ها به حال خود رها می‌شوند تا ساختارهای جالب موجود در میان داده‌ها را کشف کنند.



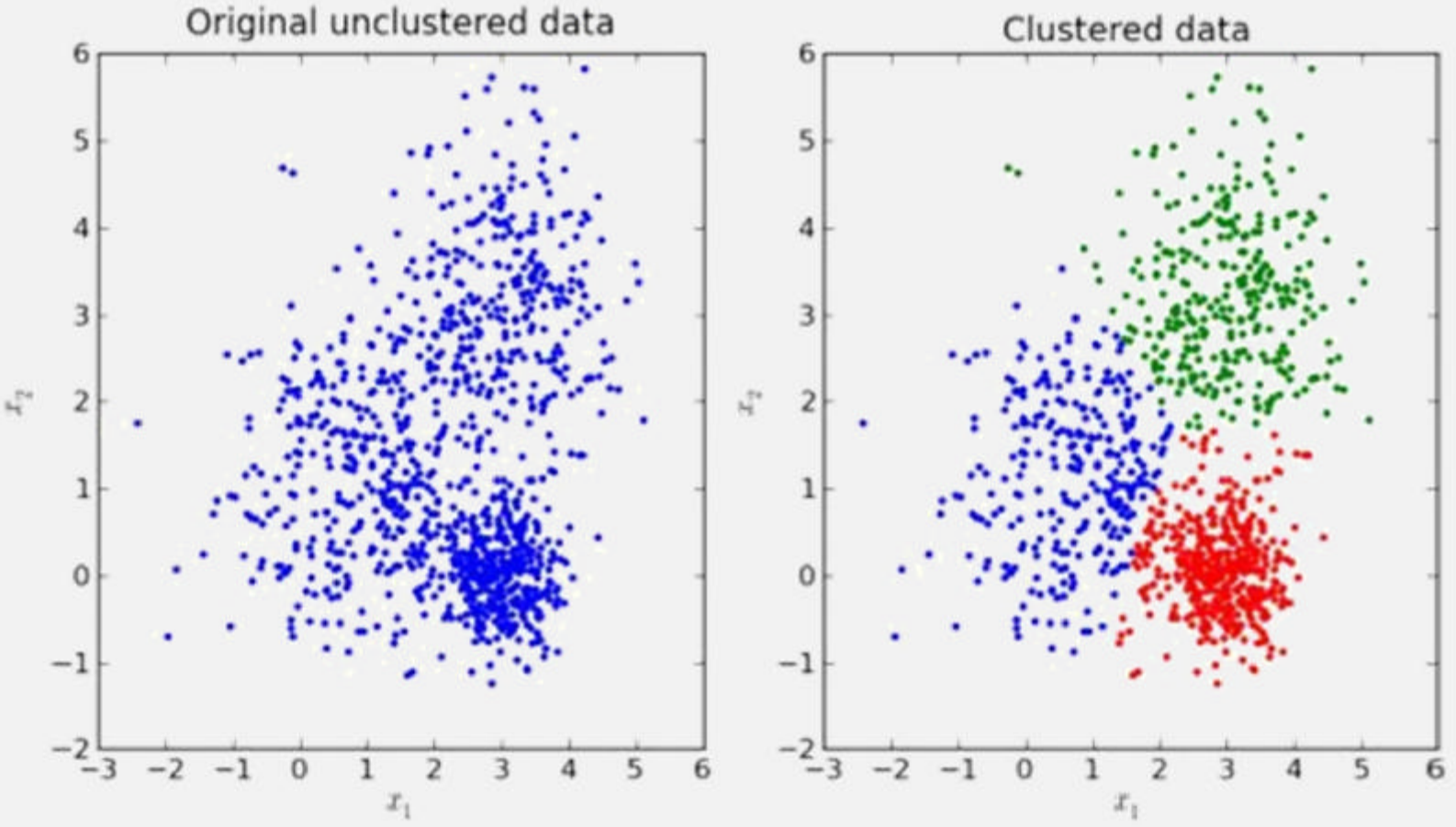
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثالی از کاربرد آموزش نظارت نشده

در این مطلب، از «مجموعه داده گل زنبق» (Iris flower data set) یا «مجموعه داده زنبق فیشر» (Fisher's Iris data set) برای انجام پیش‌بینی‌های سریع استفاده می‌شود. مجموعه داده مذکور شامل یک مجموعه از ۱۵۰ رکورد و ۵ ویژگی (مشخصه | attribute) است. این ویژگی‌ها «طول گلبرگ» (Petal Length)، «عرض گلبرگ» (Petal Width)، «طول کاسبرگ» (Sepal Length)، «عرض کاسبرگ» (Sepal width) و برچسب‌ها هستند. برچسب‌ها در این مجموعه داده در واقع تعیین می‌کنند که هر نمونه با طول کاسبرگ و گلبرگ خود از کدام گونه گل زنبق است. گونه‌های موجود در این مجموعه داده شامل «زنبق سِتوسا» (Iris Setosa)، «زنبق ویرجینیکا» (Iris Virginica) و «زنبق وِرسیکالر» (Iris Versicolor) هستند. برای الگوریتم یادگیری نظارت نشده، چهار ویژگی گل زنبق به مدل داده می‌شود و مدل پیش‌بینی می‌کند که هر نمونه به کدام دسته تعلق دارد.

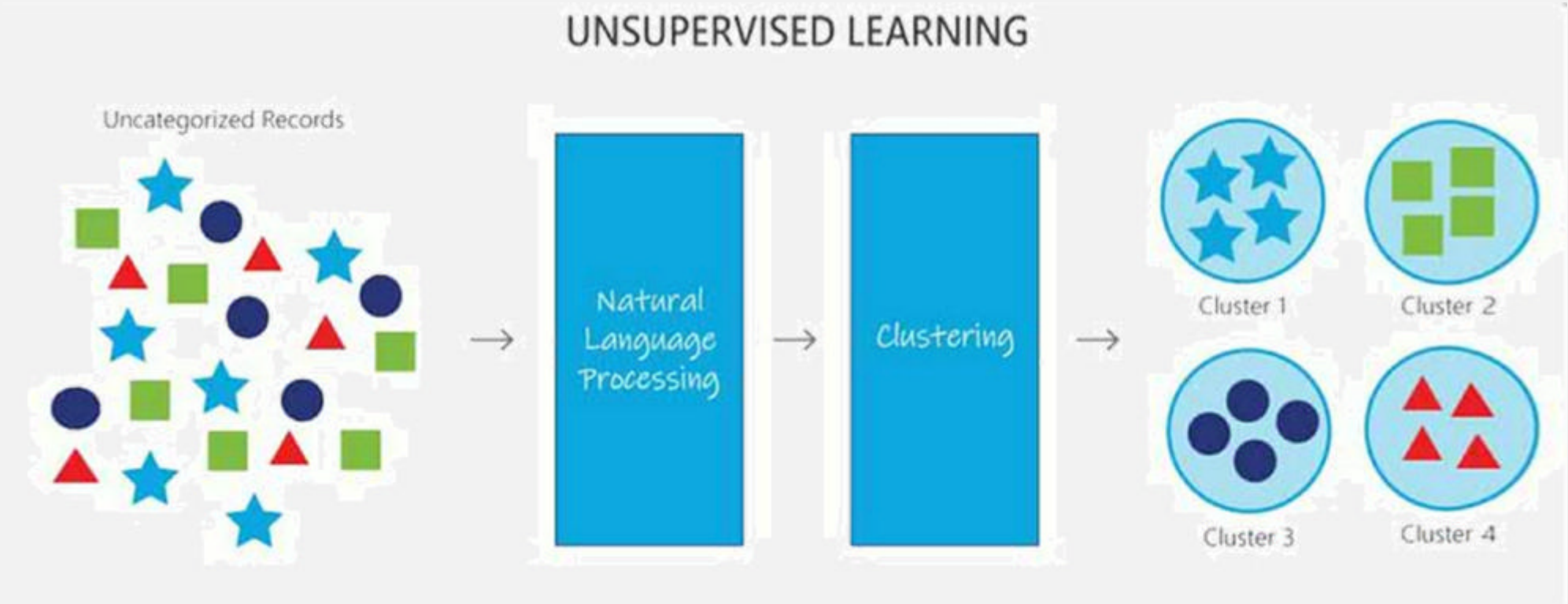
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثالی از کاربرد آموزش نظارت نشده



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

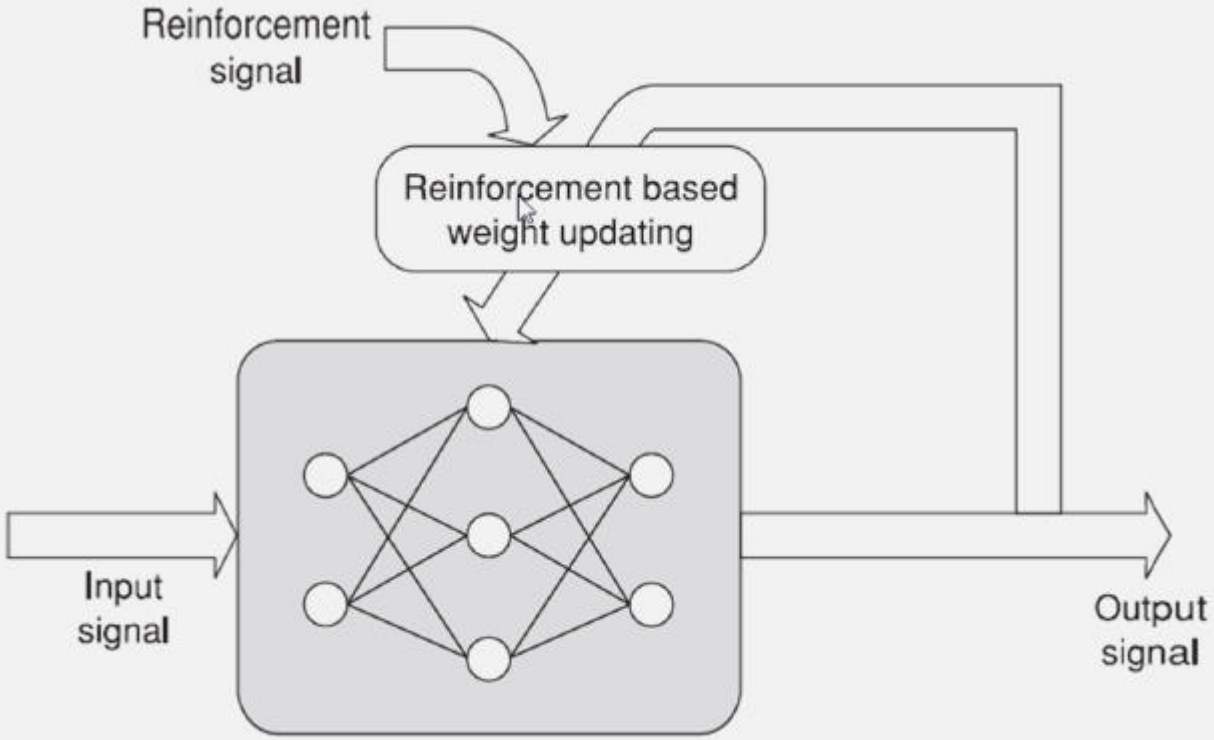
مثالی دیگر از کاربرد آموزش نظارت نشده



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش تقویتی

- یادگیری تقویتی نوعی نگاه در یادگیری ماشین است که موضوع آن طراحی الگوریتم‌هایی برای شناخت محیط و تصمیم‌گیری بهینه به منظور بیشینه کردن مجموعی از پاداش‌های دریافتی است.



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مهمترین معماری‌های شبکه‌های عصبی مصنوعی برای کاربرد در مهندسی عمران

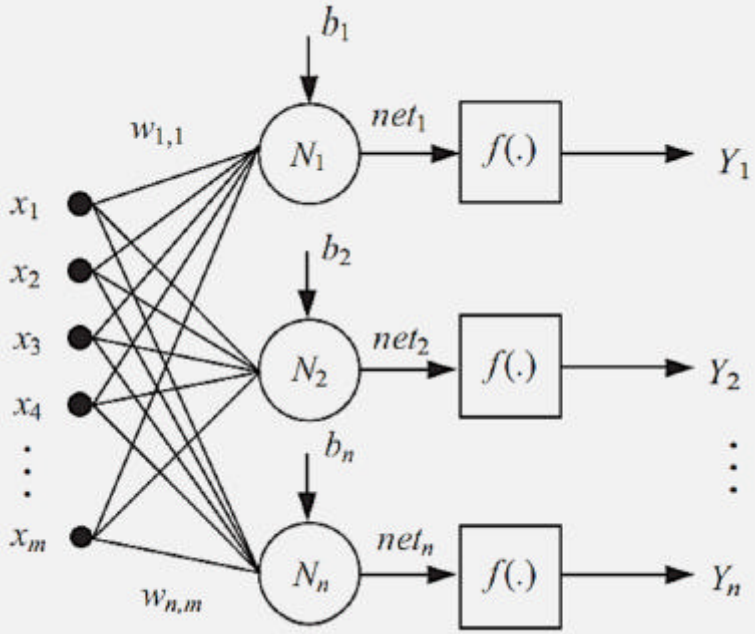
- شبکه عصبی پرسپترون چند لایه (Multilayer Perceptron)
- شبکه عصبی تابع پایه شعاعی (Radial Basis Function Neural Network)
- شبکه عصبی رگرسیون عمومی (Generalized Regression Neural Network)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه هاي عصبي پرسپترون چند لايه

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه عصبی پرسپترون یک لایه

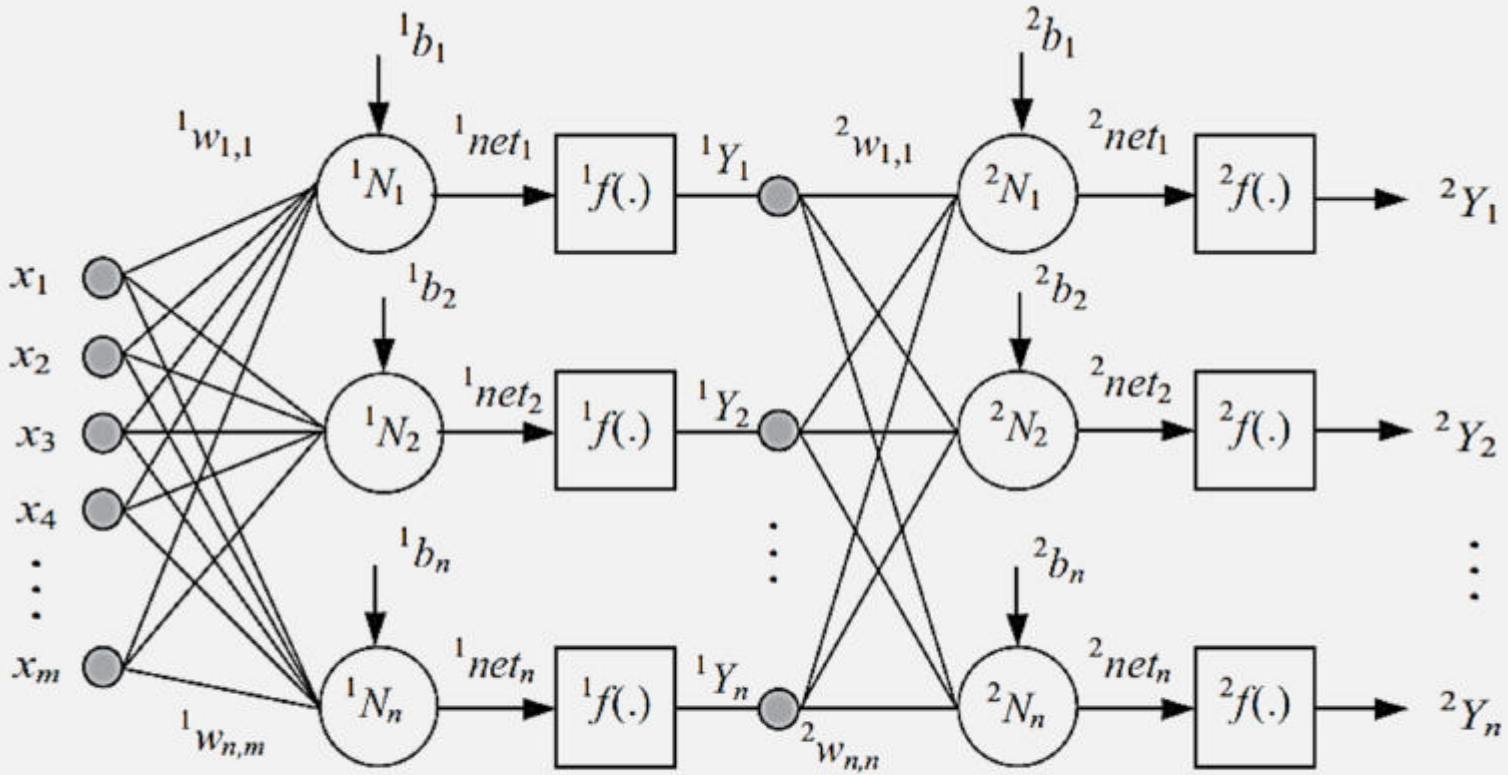


$$Y = f(W \cdot x + b)$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \qquad W = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,m} \\ w_{2,1} & w_{2,2} & \cdots & \\ \vdots & \vdots & \vdots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,m} \end{bmatrix} \qquad b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه عصبی پرسپترون چندلایه

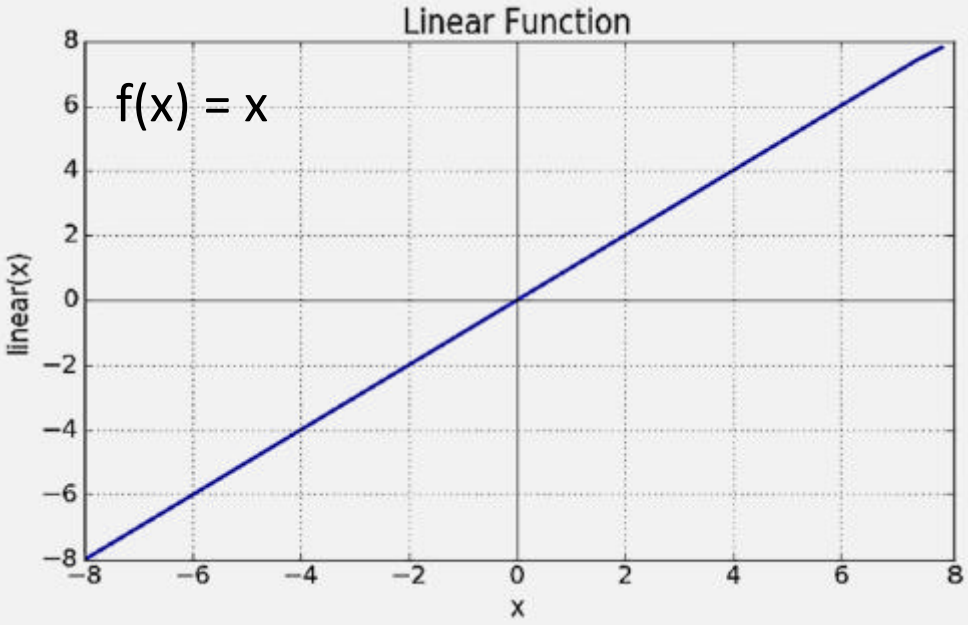


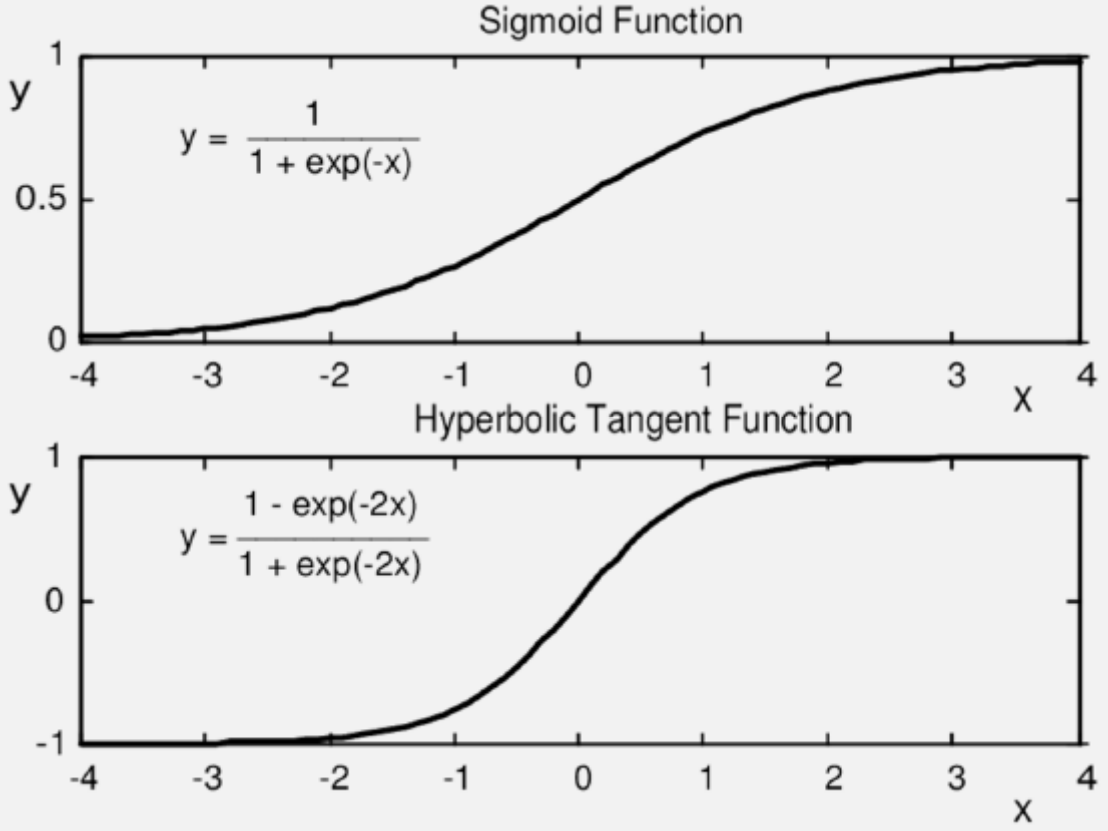
$$^1Y = {}^1f(^1W \cdot X + {}^1b) \qquad ^2Y = {}^2f(^2W \cdot {}^1Y + {}^2b)$$

لایه ورودی | لایه مخفی | لایه خروجی

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

توابع فعالسازی





Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

قواعد مختلف جهت آموزش نظارت شده شبکه عصبی

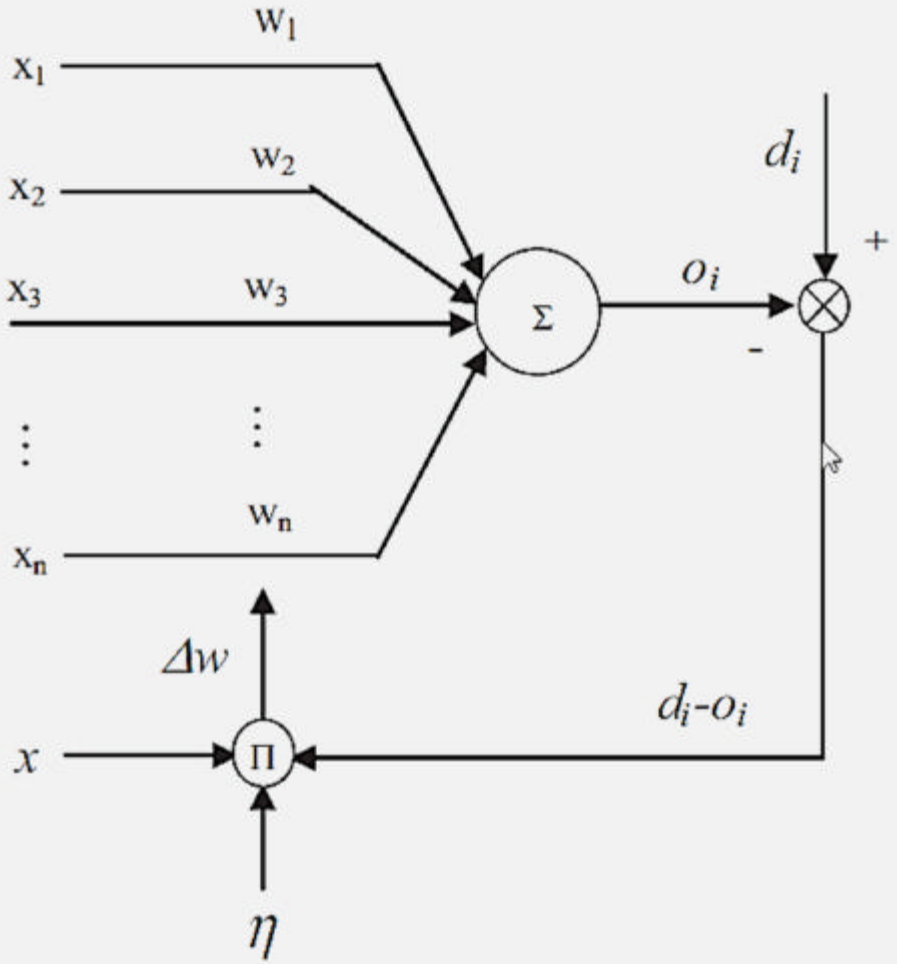
- **Widrow–Hoff rule,**
- Gradient descent,
- Delta rule,
- **Backpropagation rule,**
- Cohen–Grossberg learning rule, and
- Adaptive conjugate gradient model of Adeli and Hung.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش به روش

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

Widrow–Hoff Learning Algorithm



$$\Delta w_i = \eta(d_i - o_i)x$$

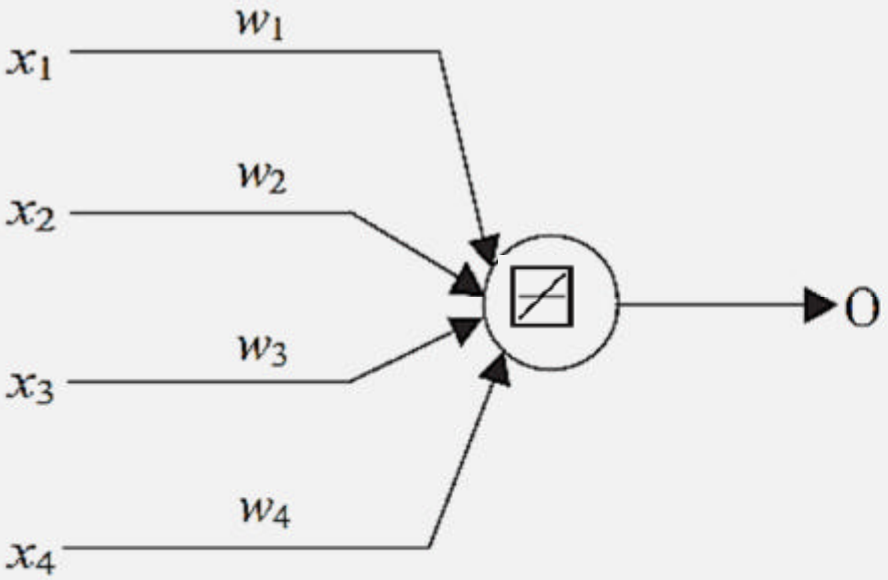
$$o_i = f(net_i) = net_i$$

$$net_i = \sum w_i^t x$$

η is the learning constant
x is the input vector
d is desired Output
O is network output

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

Widrow–Hoff Learning Algorithm-Example



$$w^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}, \; x^1 = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}, \; x^2 = \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} \text{ and } x^3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

$d = [\ 1\ -1\ 0]$

$d^1=1$

$d^2=-1$

$d^3=0$

$\eta = 1$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

Widrow–Hoff Learning Algorithm-Example

تکرار اول با استفاده از داده اول

$$net^1 = w^1 x^1 = \begin{bmatrix} 1 & -1 & 0 & .5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = 3$$

$$o^1 = f(net^1) = net^1 = 3$$

$$\Delta w^1 = \eta(d^1 - o^1)x^1 = 1 * (1 - 3) \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} -2 \\ 4 \\ -3 \\ 0 \end{bmatrix}$$

$$w^2 = w^1 + \Delta w^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + \begin{bmatrix} -2 \\ 4 \\ -3 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ 3 \\ -3 \\ 0.5 \end{bmatrix}$$

تکرار دوم و سوم را برای سایر داده‌ها انجام دهید.

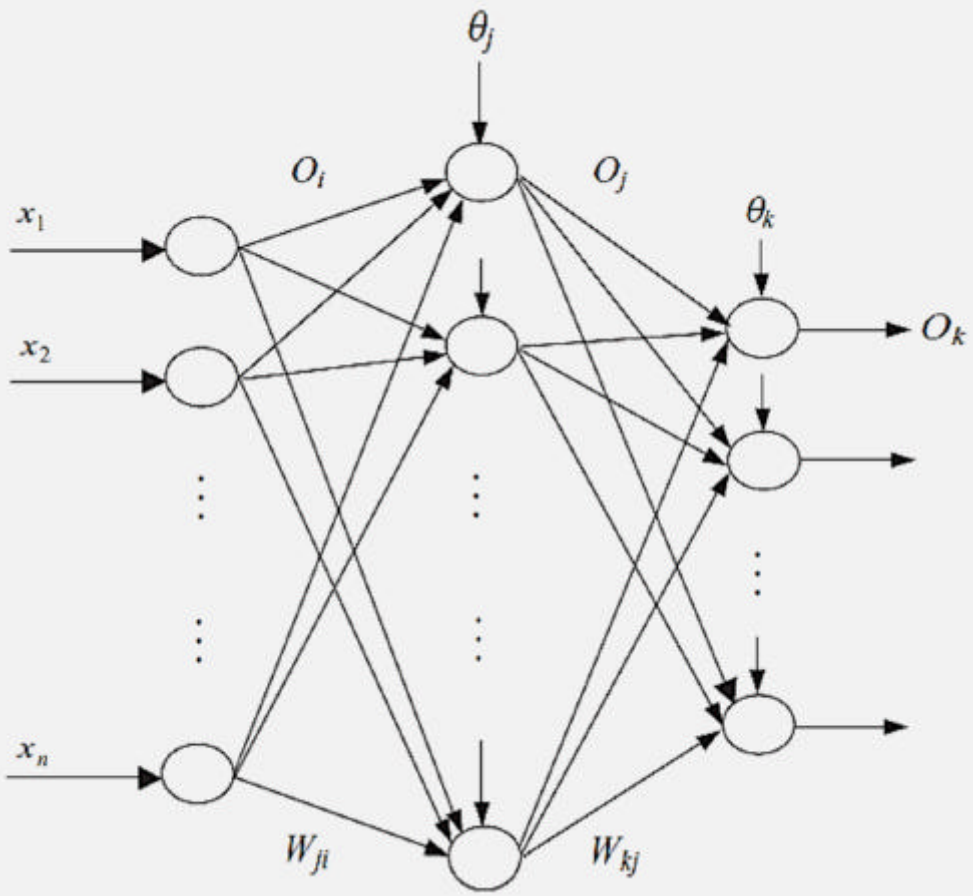
به انجام این عمل برای کلیه داده ها یک اپک آموزش گفته می‌شود.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش به روش پس انتشار

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم آموزش پس انتشار (Back-Propagation)



$$O_j = f(net_j)$$

$$net_j = \sum_i W_{ji} O_i + \theta_j$$

$$O_k = f(net_k)$$

$$net_k = \sum_j W_{kj} O_j + \theta_k$$

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

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم آموزش پس انتشار (Back-Propagation)

$$E = \frac{1}{2}\sum e^2 = \frac{1}{2}\sum_k (t_k - O_k)^2$$

Calculation of the output layer weight change

$$\Delta w_{kj} = -\eta \frac{\partial E}{\partial w_{kj}}$$

where η is the learning rate and $\eta > 0$, Δw_{kj} is the weight change and $\Delta w_{kj} = w_{kj}^{new} - w_{kj}^{old}$.

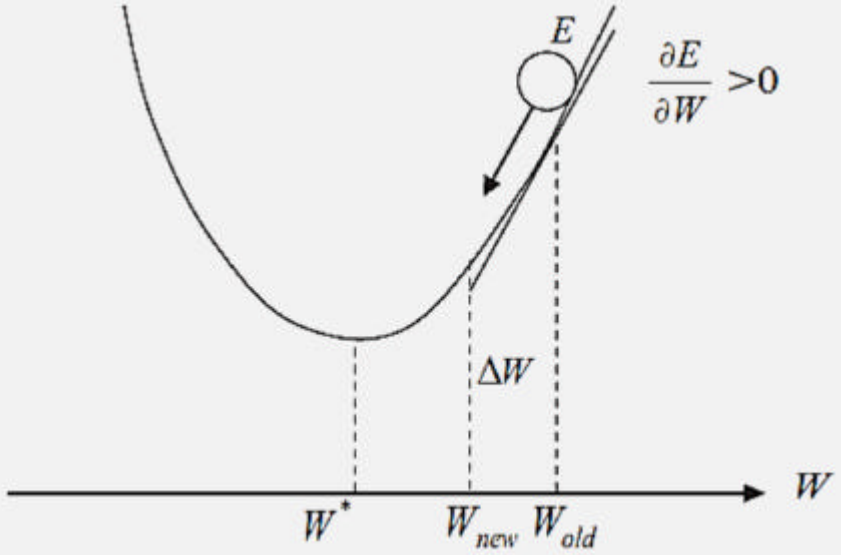
Using the chain rule, we get

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial net_k} \cdot \frac{\partial net_k}{\partial w_{kj}} = -\delta_k \cdot \frac{\partial net_k}{\partial w_{kj}}$$

where $\delta_k = -\frac{\partial E}{\partial net_k}$, termed the generalized error signal.

Now, we want to derive the term $\frac{\partial net_k}{\partial w_{kj}}$:

$$\frac{\partial net_k}{\partial w_{kj}} = \frac{\partial\left(\sum_j w_{kj} O_j + \theta_k\right)}{\partial w_{kj}} = O_j$$



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم آموزش پس انتشار (Back-Propagation)

To compute the term δ_k, we apply the chain rule

$$\delta_k = -\frac{\partial E}{\partial net_k} = -\frac{\partial E}{\partial O_k}\frac{\partial O_k}{\partial net_k}$$

$$\frac{\partial E}{\partial O_k} = \frac{\frac{1}{2}\sum_k (t_k - O_k)^2}{\partial O_k} = -(t_k - O_k)$$

$$\frac{\partial O_k}{\partial net_k} = f'(net_k)$$

Note. $f'(x)$ denotes the derivative of $f(x)$ with respect to x and can easily be derived as follows:

$$f'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1}{(1+e^{-x})}\left(1 - \frac{1}{1+e^{-x}}\right)$$

$$f'(x) = f(x)(1 - f(x))$$

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

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم آموزش پس انتشار (Back-Propagation)

Hence,

$$\frac{\partial O_k}{\partial net_k} = O_k(1 - O_k)$$

Thus, we have

$$\Delta w_{kj} = \eta \delta_k O_j$$

$$\delta_k = O_k(1 - O_k)(t_k - O_k)$$

Similarly, we can calculate the bias change

$$\Delta\theta_k = -\eta \frac{\partial E}{\partial \theta_k} = \eta \left(-\frac{\partial E}{\partial net_k} \frac{\partial net_k}{\partial \theta_k} \right) = \eta \delta_k \frac{\partial \left(\sum_j w_{kj} O_j + \theta_k \right)}{\partial \theta_k} = \eta \delta_k$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم آموزش پس انتشار (Back-Propagation)

Calculation of the hidden-layer weight change

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}}$$

Using the chain rule we get

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial net_j} \cdot \frac{\partial net_j}{\partial w_{ji}} = -\delta_j \cdot \frac{\partial net_j}{\partial w_{ji}}$$

Now, we get

$$\frac{\partial net_j}{\partial w_{ji}} = O_i$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم آموزش پس انتشار (Back-Propagation)

Using the chain rule, we get

$$\delta_j = -\frac{\partial E}{\partial net_j} = -\sum_k \frac{\partial E}{\partial net_k}\frac{\partial net_k}{\partial O_j}\frac{\partial O_j}{\partial net_j}$$

$$\delta_j = \sum_k \delta_k w_{kj} f'(net_j)$$

$$\delta_j = O_j(1 - O_j)\sum_k \delta_k w_{kj}$$

Thus, we have

$$\Delta w_{ji} = \eta \delta_j O_i$$

and the bias change

$$\Delta\theta_j = -\eta\frac{\partial E}{\partial \theta_j} = \eta\delta_j$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مراحل آموزش شبکه عصبی پرسپترون چند لایه با استفاده از روش پس انتشار

1. Initialize w_{kj}, w_{ji}, θ_k and θ_j and set learning rate η.
2. Propagate inputs to network and calculate O_j, O_k.
3. Calculate δ_k by the formula

$$\delta_k = O_k(1 - O_k)(t_k - O_k)$$

4. Calculate change of weights and biases by

$$\Delta w_{kj} = \eta \delta_k O_j$$
$$\Delta \theta_k = \eta \delta_k$$

5. Calculate δ_j by the formula

$$\delta_j = O_j(1 - O_j)\sum_k \delta_k w_{kj}$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مراحل آموزش شبکه عصبی پرسپترون چند لایه با استفاده از روش پس انتشار

6. Calculate change of weights and biases by

$$\Delta w_{ji} = \eta \delta_j O_i$$
$$\Delta \theta_j = \eta \delta_j$$

7. Calculate new weights and biases

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}$$
$$\theta_j(t+1) = \theta_j(t) + \Delta \theta_j$$
$$w_{kj}(t+1) = w_{kj}(t) + \Delta w_{kj}$$
$$\theta_k(t+1) = \theta_k(t) + \Delta \theta_k$$

8. Set $t \leftarrow t+1$ and go to step 2.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش به روش پس انتشار–شبکه عصبی با تابع انتقال سیگموئید و خطی

تابع انتقال سیگموئید در لایه پنهان

$$f'(x)=\frac{e^{-x}}{(1+e^{-x})^2}=\frac{1}{(1+e^{-x})}(1-\frac{1}{1+e^{-x}})=f(x)(1-f(x))$$

تابع انتقال خطی در لایه خروجی

$$f'(x)=1$$

- برای جمله خطای لایه آخر:

$$\delta_k=(t_k-O_k)$$

- برای جملات خطای لایه‌های پنهان:

$$\delta_j=O_j(1-O_j)\sum_k \delta_j w_{kj}$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش به روش پس انتشار–شبکه عصبی با تابع انتقال سیگموئید و خطی

تابع انتقال سیگموئید در لایه پنهان

$$f'(x)=\frac{e^{-x}}{(1+e^{-x})^2}=\frac{1}{(1+e^{-x})}(1-\frac{1}{1+e^{-x}})=f(x)(1-f(x))$$

تابع انتقال خطی در لایه خروجی

$$f'(x)=1$$

- برای جمله خطای لایه آخر:

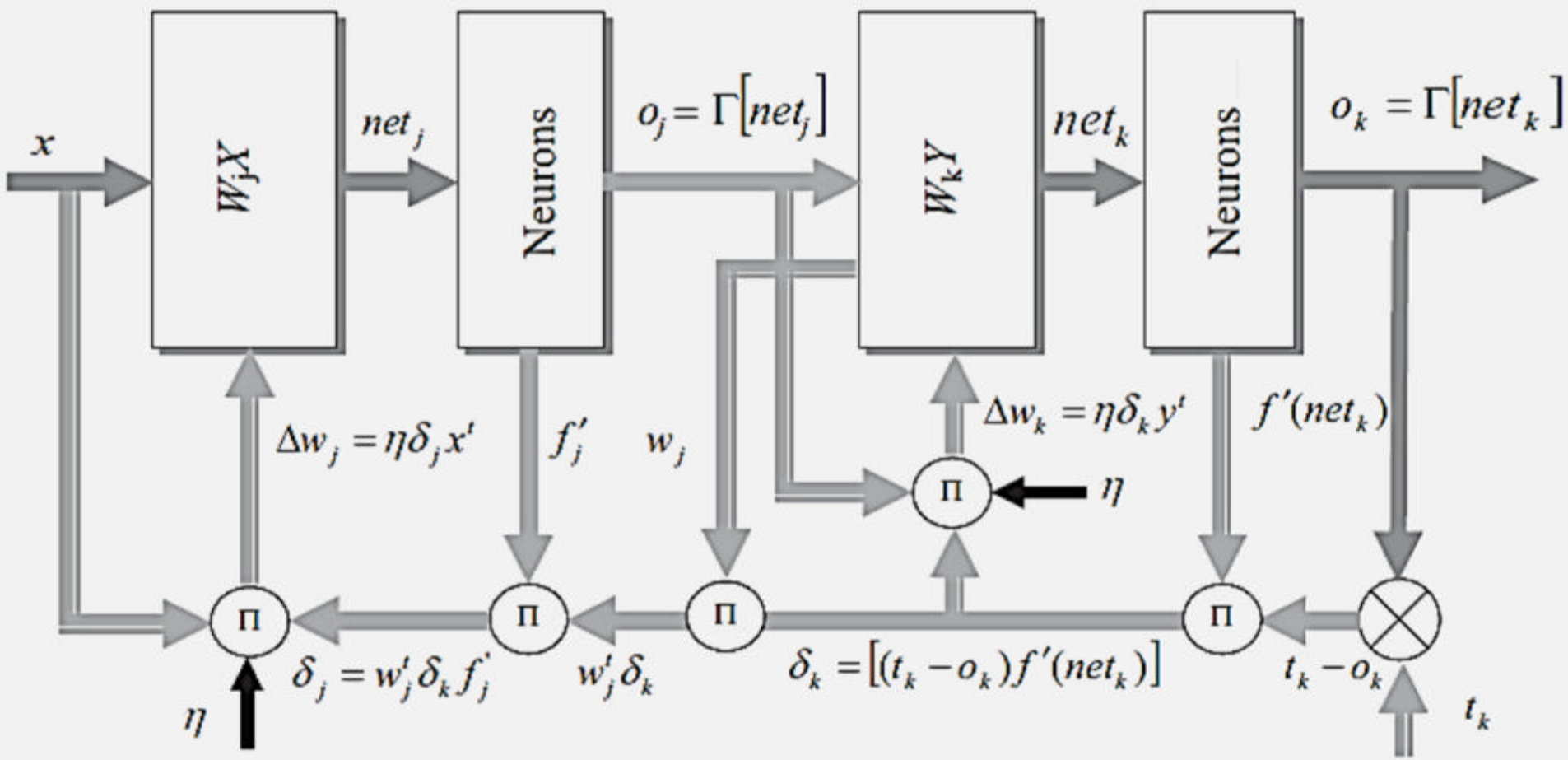
$$\delta_k=(t_k-O_k)$$

- برای جملات خطای لایه‌های پنهان:

$$\delta_j=O_j(1-O_j)\sum_k \delta_j w_{kj}$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

فرایند آموزش شبکه عصبی پرسپترون چند لایه با استفاده از روش پس انتشار



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش Stochastic

- در فرایند آموزش شبکه های عصبی با استفاده از یک سمپل خیلی بزرگ معمولا از روش Online learning استفاده می شود.
- در این روش از همه داده های سمپل برای آپدیت کردن وزن ها استفاده نمی شود بلکه نمونه های آموزشی یکی یکی به شبکه عصبی وارد شده و وزن های شبکه عصبی پس از ورود هر نمونه آموزشی آپدیت می شود. یک مرحله عبور از تمام نمونه های آموزشی Epoch گفته می شود.
- در این روش، تابع هزینه به جای محاسبه بر روی تمام نمونه های آموزشی (E)، فقط روی یک نمونه آموزشی (E_i) محاسبه می شود.

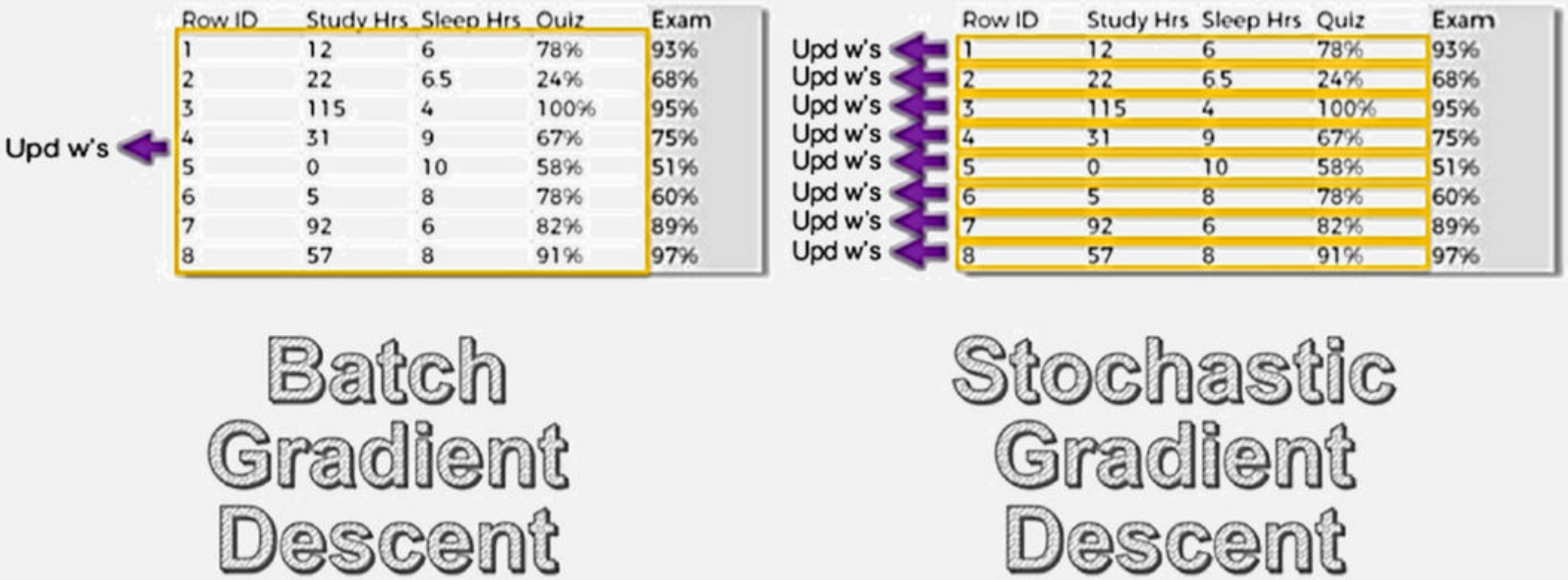
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش Batch

- در این روش، برای هربار آپدیت کردن وزن ها باید از تمام نمونه های آموزشی استفاده کرد.
- در نتیجه برای اجرای هر گام از این الگوریتم نیازمند محاسبات بسیار گسترده ای می باشیم.
- این مساله هنگامی که با یک سمپل داده بسیار بزرگ سر و کار داریم کاملا مشهود خواهد بود. علاوه بر این اگر تابع هزینه دارای چندین مینیمم محلی باشد آنگاه تضمینی برای رسیدن به مینیمم مطلق در این روش وجود ندارد.

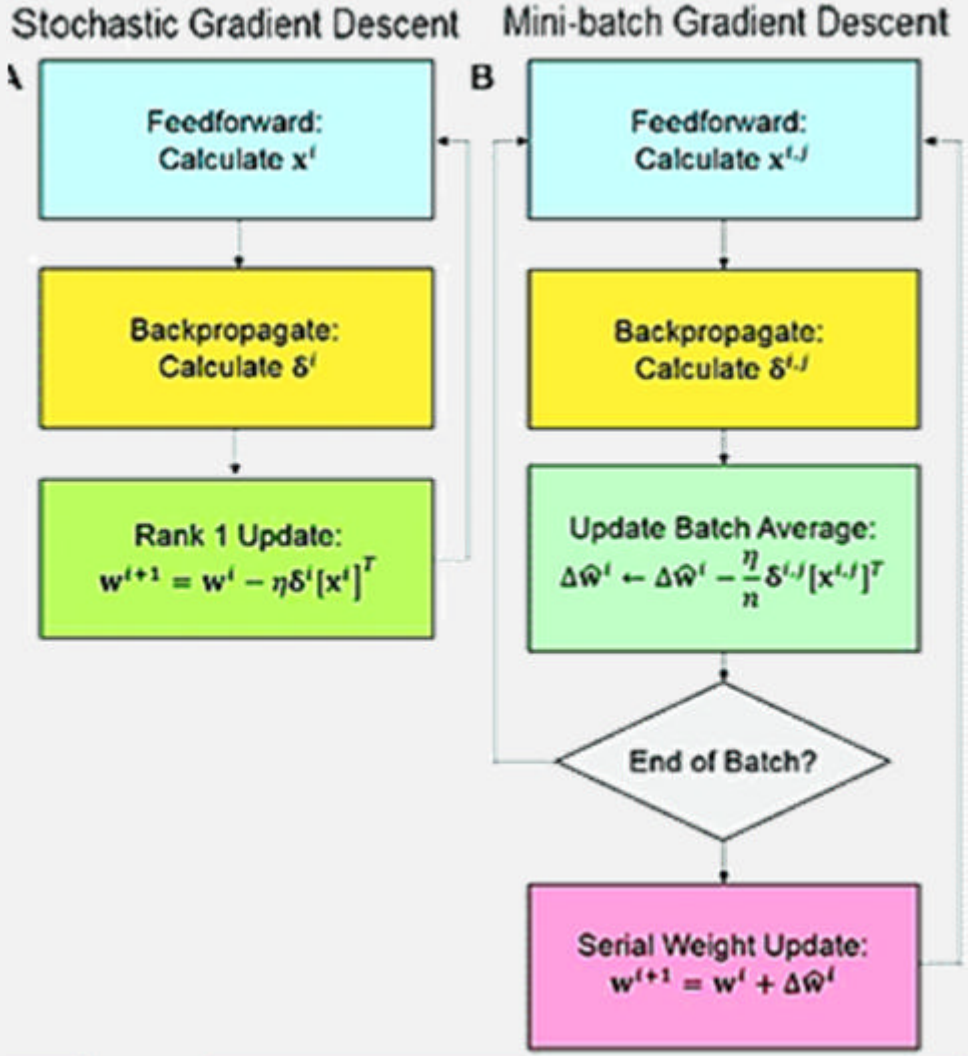
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مقایسه روش Batch و Stochastic



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مقایسه روش Batch و Stochastic



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

ترم مومنتم برای افزایش سرعت و پایداری آموزش شبکه

- می توان یک ترم **Momentum** به فرمول آپدیت وزن های شبکه عصبی اضافه کرد. با اضافه نمودن این ترم، درصدی از مقدار آپدیت قبلی وزن ها نیز به وزن فعلی اضافه می شود، در نتیجه از همگرایی به مینیم های محلی کوچک جلوگیری شده و الگوریتم از احتمال بیشتری برای همگرایی به مینیمم مطلق برخوردار می‌شود.
- در روش **Batch** شاخص **t-1** مربوط به وزن ها در **Epoch** قبلی و در روش **Stochastic** مربوط به تکرار قبلی می باشد. γ معمولا بین ۰/۵ و ۱ انتخاب می شود.
- این روش برای **Online learning** به خوبی عمل می کند. در این حالت روش گرادیان نزولی در یک منحنی صاف و با سرعت بیشتری همگرایی می رسد.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

ترم مومنتم برای افزایش سرعت و پایداری آموزش شبکه

$$\Delta w_{ij} = (\eta * \frac{\partial E}{\partial w_{ij}})$$

weight increment — learning rate — weight gradient

- ✓ پارامتر مومنتم γ عددی بین ۰ تا ۱ خواهد بود.
- ✓ این پارامتر معمولاً بین ۰/۵ و ۱ در نظر گرفته خواهد شد.

$$\Delta w_{ij} = (\eta * \frac{\partial E}{\partial w_{ij}}) + (\gamma * \Delta w_{ij}^{t-1})$$

momentum factor — weight increment, previous iteration

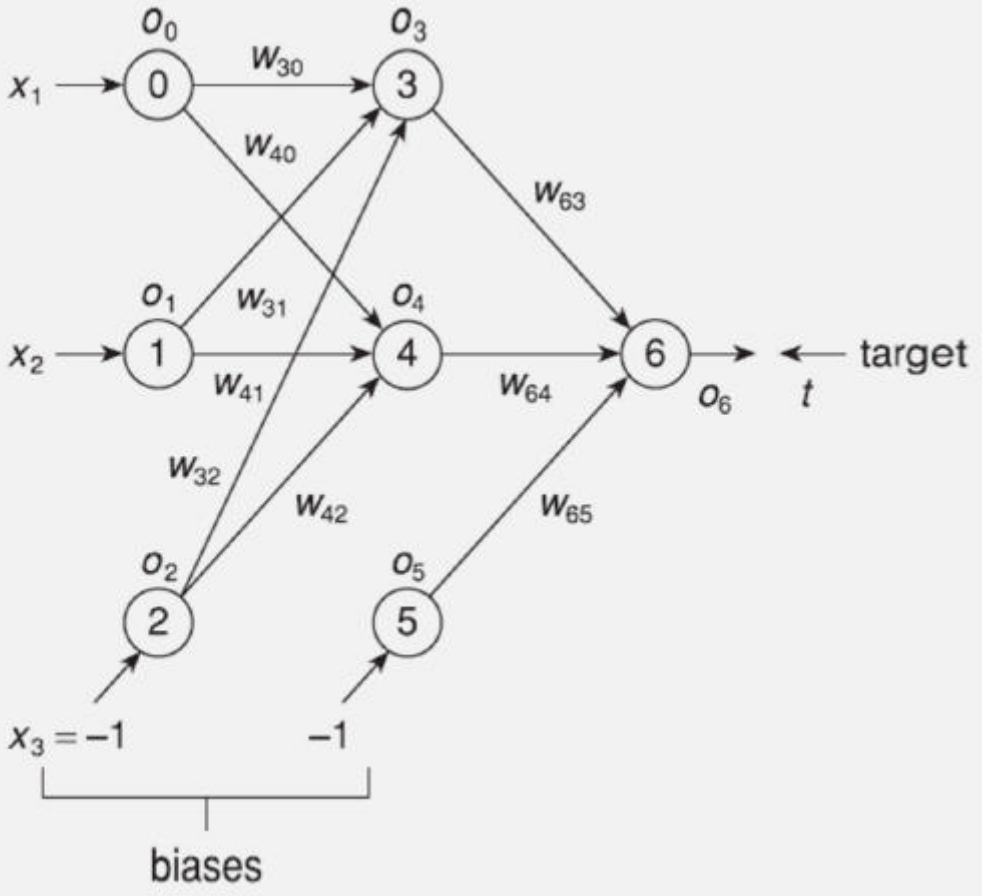
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال اول آموزش به روش پس انتشار

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

$$\mathbf{x}^{(1)} = (0.3, 0.4), \quad \mathbf{t}^{(1)} = (0.88),$$
$$\mathbf{x}^{(2)} = (0.1, 0.6), \quad \mathbf{t}^{(2)} = (0.82),$$
$$\mathbf{x}^{(3)} = (0.9, 0.4), \quad \mathbf{t}^{(3)} = (0.57),$$



$$o = f(\text{tot}) = \frac{1}{1 + e^{-\lambda \text{tot}}},$$

using $\lambda = 1$, then $f'(\text{tot}) = o(1 - o)$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Step (1) – Initialization

- Initialize the weights to small random values. We assume all weights are initialized to 0.2; set learning rate to $\eta = 0.2$; set maximum tolerable error to $E_{max} = 0.01$ (i.e., 1% error); set current error value to $E = 0$; set current training pattern to $k = 1$.

Training Loop – Loop (1)

Step (2) – Apply input pattern

- Apply the 1st input pattern to the input layer:

$$\mathbf{x}^{(1)} = (0.3, 0.4),\ \mathbf{t}^{(1)} = (0.88),\ \text{then},\ o_0 = \mathbf{x}_1 = 0.3;\ o_1 = \mathbf{x}_2 = 0.4;\ o_2 = \mathbf{x}_3 = -1$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Step (3) – Forward propagation

- Propagate the signal forward through the network:

$$o_3 = f(w_{30}o_0 + w_{31}o_1 + w_{32}o_2) = 0.4850$$
$$o_4 = f(w_{40}o_0 + w_{41}o_1 + w_{42}o_2) = 0.4850$$
$$o_5 = -1$$
$$o_6 = f(w_{63}o_3 + w_{64}o_4 + w_{65}o_5) = 0.4985$$

Step (4) – Output error measure

- Compute the error value E and the error signal δ_6 of the output layer:

$$E = \tfrac{1}{2}(t - o_6)^2 + E = 0.0728$$
$$\delta_6 = f'(\text{tot}_6)(t - o_6)$$
$$= o_6(1 - o_6)(t - o_6)$$
$$= 0.0954$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Step (5) – Error backpropagation

- Propagate the errors backward to update the weights and compute the error signals of the preceding layers.

Third layer weight updates:

$$\Delta w_{63} = \eta \delta_6 o_3 = 0.0093 \qquad w_{63}^{new} = w_{63}^{old} + \Delta w_{63} = 0.2093$$

$$\Delta w_{64} = \eta \delta_6 o_4 = 0.0093 \qquad w_{64}^{new} = w_{64}^{old} + \Delta w_{64} = 0.2093$$

$$\Delta w_{65} = \eta \delta_6 o_5 = -0.0191 \qquad w_{65}^{new} = w_{65}^{old} + \Delta w_{65} = 0.1809$$

Second layer error signals:

$$\delta_3 = f_3'(\text{tot}_3) \sum_{i=6}^{6} w_{i3}\delta_i = o_3(1 - o_3) w_{63}\delta_6 = 0.0048$$

$$\delta_4 = f_4'(\text{tot}_4) \sum_{i=6}^{6} w_{i4}\delta_i = o_4(1 - o_4) w_{64}\delta_6 = 0.0048$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Second layer weight updates:

$$\Delta w_{30} = \eta \delta_3 o_0 = 0.00028586 \qquad w_{30}^{new} = w_{30}^{old} + \Delta w_{30} = 0.2003$$
$$\Delta w_{31} = \eta \delta_3 o_1 = 0.00038115 \qquad w_{31}^{new} = w_{31}^{old} + \Delta w_{31} = 0.2004$$
$$\Delta w_{32} = \eta \delta_3 o_2 = -0.00095288 \qquad w_{32}^{new} = w_{32}^{old} + \Delta w_{32} = 0.1990$$

$$\Delta w_{40} = \eta \delta_4 o_0 = 0.00028586 \qquad w_{40}^{new} = w_{40}^{old} + \Delta w_{40} = 0.2003$$
$$\Delta w_{41} = \eta \delta_4 o_1 = 0.00038115 \qquad w_{41}^{new} = w_{41}^{old} + \Delta w_{41} = 0.2004$$
$$\Delta w_{42} = \eta \delta_4 o_2 = -0.00095288 \qquad w_{42}^{new} = w_{42}^{old} + \Delta w_{42} = 0.1990$$

Training Loop – Loop (2)

Step (2) – Apply the 2nd input pattern

Apply the 2nd input pattern to the input layer:

$\mathbf{x}^{(2)} = (0.1, 0.6)$, $\mathbf{t}^{(2)} = (0.82)$, then, $o_0 = 0.1$, $o_1 = 0.6$, $o_2 = -1$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Step (3) – Forward propagation

$$o_3 = f(w_{30}o_0 + w_{31}o_1 + w_{32}o_2) = 0.4853$$

$$o_4 = f(w_{40}o_0 + w_{41}o_1 + w_{42}o_2) = 0.4853$$

$$o_5 = -1$$

$$o_6 = f(w_{63}o_3 + w_{64}o_4 + w_{65}o_5) = 0.5055$$

Step (4) – Output error measure

$$E = \tfrac{1}{2}(t - o_6)^2 + E = 0.1222$$

$$\delta_6 = o_6(1 - o_6)(t - o_6) = 0.0786$$

Step (5) – Error backpropagation

Third layer weight updates:

$$\Delta w_{63} = \eta\delta_6 o_3 = 0.0076 \qquad w_{63}^{new} = w_{63}^{old} + \Delta w_{63} = 0.2169$$

$$\Delta w_{64} = \eta\delta_6 o_4 = 0.0076 \qquad w_{64}^{new} = w_{64}^{old} + \Delta w_{64} = 0.2169$$

$$\Delta w_{65} = \eta\delta_6 o_5 = -0.0157 \qquad w_{65}^{new} = w_{65}^{old} + \Delta w_{65} = 0.1652$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Second layer error signals:

$$\delta_3 = f'_3(\text{tot}_3)\sum_{i=6}^{6} w_{i3}\delta_i = o_3(1-o_3)w_{63}\delta_6 = 0.0041$$

$$\delta_4 = f'_4(\text{tot}_4)\sum_{i=6}^{6} w_{i4}\delta_i = o_4(1-o_4)w_{64}\delta_6 = 0.0041$$

Second layer weight updates:

$$\Delta w_{30} = \eta\delta_3 o_0 = 0.000082169 \qquad w_{30}^{new} = w_{30}^{old} + \Delta w_{30} = 0.2004$$

$$\Delta w_{31} = \eta\delta_3 o_1 = 0.00049302 \qquad w_{31}^{new} = w_{31}^{old} + \Delta w_{31} = 0.2009$$

$$\Delta w_{32} = \eta\delta_3 o_2 = -0.00082169 \qquad w_{32}^{new} = w_{32}^{old} + \Delta w_{32} = 0.1982$$

$$\Delta w_{40} = \eta\delta_4 o_0 = 0.000082169 \qquad w_{40}^{new} = w_{40}^{old} + \Delta w_{40} = 0.2004$$

$$\Delta w_{41} = \eta\delta_4 o_1 = 0.00049302 \qquad w_{41}^{new} = w_{41}^{old} + \Delta w_{41} = 0.2009$$

$$\Delta w_{42} = \eta\delta_4 o_2 = -0.00082169 \qquad w_{42}^{new} = w_{42}^{old} + \Delta w_{42} = 0.1982$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Training Loop – Loop (3)

Step (2) – Apply the 3rd input pattern to the input layer

$\mathbf{x}^{(2)} = (0.9, 0.4)$, $\mathbf{t}^{(2)} = (0.57)$, then, $o_0 = 0.9$, $o_1 = 0.4$, $o_2 = -1$

Step (3) – Forward propagation

$$o_3 = f(w_{30}o_0 + w_{31}o_1 + w_{32}o_2) = 0.5156$$

$$o_4 = f(w_{40}o_0 + w_{41}o_1 + w_{42}o_2) = 0.5156$$

$$o_5 = -1$$

$$o_6 = f(w_{63}o_3 + w_{64}o_4 + w_{65}o_5) = 0.5146$$

Step (4) – Output error measure

$$E = \tfrac{1}{2}(t - o_6)^2 + E = 0.1237$$

$$\delta_6 = o_6(1 - o_6)(t - o_6) = 0.0138$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Step (5) – Error backpropagation

Third layer weight updates:

$$\Delta w_{63} = \eta \delta_6 o_3 = 0.0014 \qquad w_{63}^{new} = w_{63}^{old} + \Delta w_{63} = 0.2183$$

$$\Delta w_{64} = \eta \delta_6 o_4 = 0.0014 \qquad w_{64}^{new} = w_{64}^{old} + \Delta w_{64} = 0.2183$$

$$\Delta w_{65} = \eta \delta_6 o_5 = -0.0028 \qquad w_{65}^{new} = w_{65}^{old} + \Delta w_{65} = 0.1624$$

Second layer error signals:

$$\delta_3 = f_3'(\text{tot}_3) \sum_{i=6}^{6} w_{i3}\delta_i = o_3(1 - o_3) w_{63}\delta_6 = 0.00074948$$

$$\delta_4 = f_4'(\text{tot}_4) \sum_{i=6}^{6} w_{i4}\delta_i = o_4(1 - o_4) w_{64}\delta_6 = 0.00074948$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Second layer weight updates:

$$\Delta w_{30} = \eta \delta_3 o_0 = 0.00013491 \qquad w_{30}^{new} = w_{30}^{old} + \Delta w_{30} = 0.2005$$

$$\Delta w_{31} = \eta \delta_3 o_1 = 0.000059958 \qquad w_{31}^{new} = w_{31}^{old} + \Delta w_{31} = 0.2009$$

$$\Delta w_{32} = \eta \delta_3 o_2 = -0.00014990 \qquad w_{32}^{new} = w_{32}^{old} + \Delta w_{32} = 0.1981$$

$$\Delta w_{40} = \eta \delta_4 o_0 = 0.00013491 \qquad w_{40}^{new} = w_{40}^{old} + \Delta w_{40} = 0.2005$$

$$\Delta w_{41} = \eta \delta_4 o_1 = 0.000059958 \qquad w_{41}^{new} = w_{41}^{old} + \Delta w_{41} = 0.2009$$

$$\Delta w_{42} = \eta \delta_4 o_2 = -0.00014990 \qquad w_{42}^{new} = w_{42}^{old} + \Delta w_{42} = 0.1981$$

Step (6) – One-epoch looping

The training patterns have been cycled once, which is called one *epoch*, so we stop the loop and continue with step (6) below.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۱

Step (7) – Total error checking

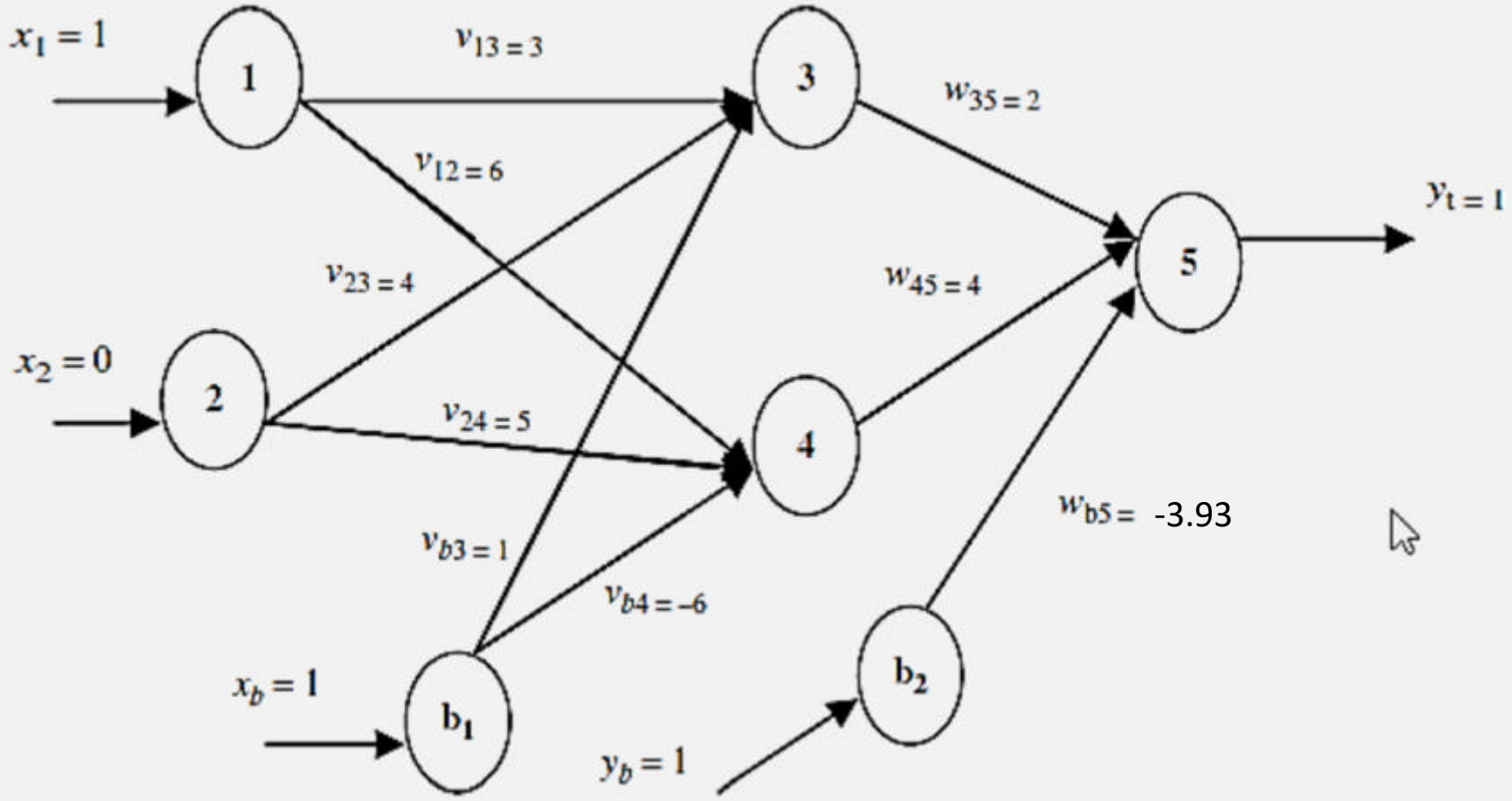
Check whether the current total error is acceptable. If $E < E_{max}$ then terminate the training process and output the final weights. Otherwise $E = 0$, $k = 1$, and initiate the new training epoch by going to Step (1). In this example, $E = 0.1237$ and $E_{max} = 0.01$, which means that we have to continue with the next epoch by cycling the training data again. We keep the training process going until the desired performance goal is met (i.e., $E < E_{max}$) or until a specified number of epochs has been reached.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال دوم آموزش به روش پس انتشار

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۲



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مثال آموزش به روش پس انتشار ۲

Let us first compute net information received by neurons 3 and 4 in the inner layer:

$$net_3 = (3)(1) + (4)(0) + (1)(1) = 4$$

$$net_4 = (1)(6) + (0)(5) + (1)(-6) = 0$$

Now, we pass the net information through the sigmoid activation function to obtain outputs from neurons 3 and 4 that are inputs for neuron 5:

$$y_3 = \frac{1}{1 + e^{-4}} = 0.982$$

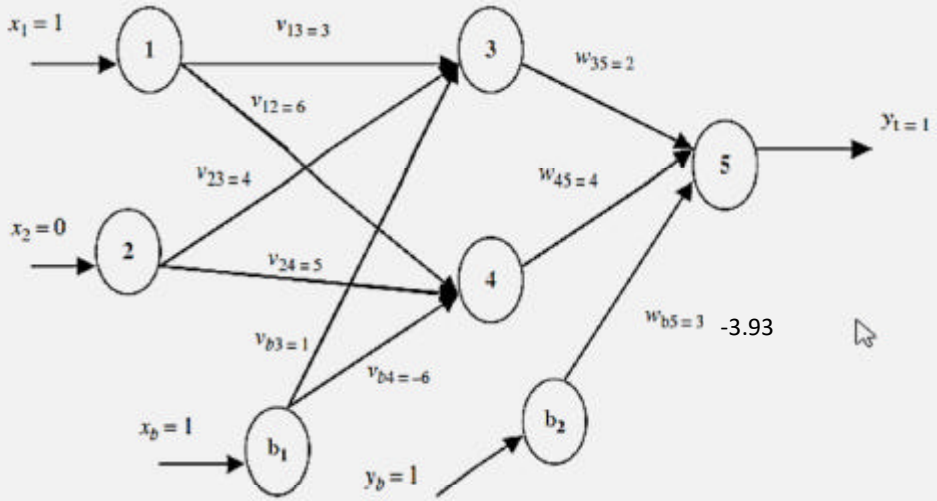
$$y_4 = \frac{1}{1 + e^{-0}} = 0.50$$

We compute the net information received by neuron 5 in the output layer as:

$$net_5 = (0.982)(2) + (0.50)(4) + (1)(-3.93) = 0.04$$

Now, we pass the net information through the activation function to obtain the model output as:

$$y_4 = \frac{1}{1 + e^{-0.04}} = 0.51$$



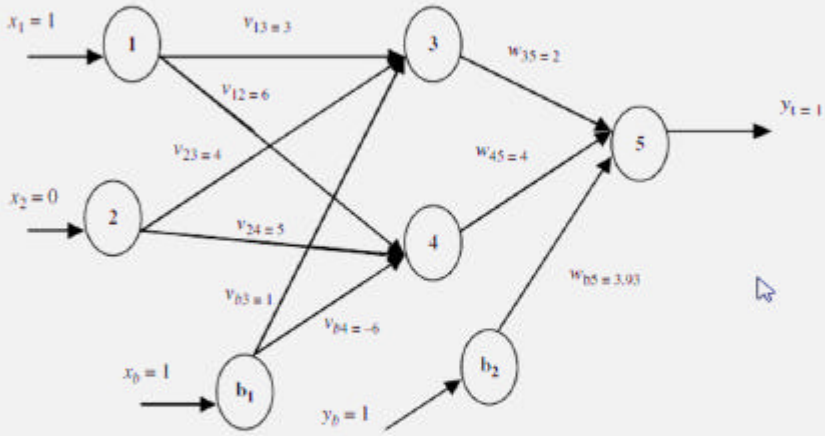
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش به روش پس انتشار ۲

We can now compute the error, which is the difference between target output and model-produced output, as

$$E = 1 - 0.51 = 0.49$$

Backward pass



First by equation (3.11):

$$\beta_{o5} = y_5(1 - y_5)(y_5 - y_t) = 0.51(1 - 0.51)(-0.49) = -0.1225$$

Then, by equation (3.22):

$$\beta_{m4} = y_4(1 - y_4)w_{45}\beta_{o5} = 0.50(1 - 0.50)(4)(-0.1225) = -0.1225$$

$$\beta_{m3} = y_3(1 - y_3)w_{35}\beta_{o5} = 0.982(1 - 0.982)(2)(-0.1225) = -0.0043$$

Then by equation (3.10), we can find new values for the connection weights between inner and output layers as:

$$w_{35}^{new} = w_{35}^{old} - \eta\beta_{o5}y_3 = 2 - (0.1)(-0.1225)(0.982) = 2.012$$

$$w_{45}^{new} = w_{45}^{old} - \eta\beta_{o5}y_4 = 4 - (0.1)(-0.1225)(0.50) = 4.012$$

$$w_{05}^{new} = w_{05}^{old} - \eta\beta_{o5}y_b = -3.93 - (0.1)(-0.1225)(1) = -3.9078$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

آموزش به روش پس انتشار ۲

Now, we can update the connection weights between inner and input layers using equation (3.21) as:

$$v_{13}^{new} = v_{13}^{old} - \eta\beta_{m3}x_1 = 3.0 - (0.1)(-0.0043)(1) = 3.0004$$

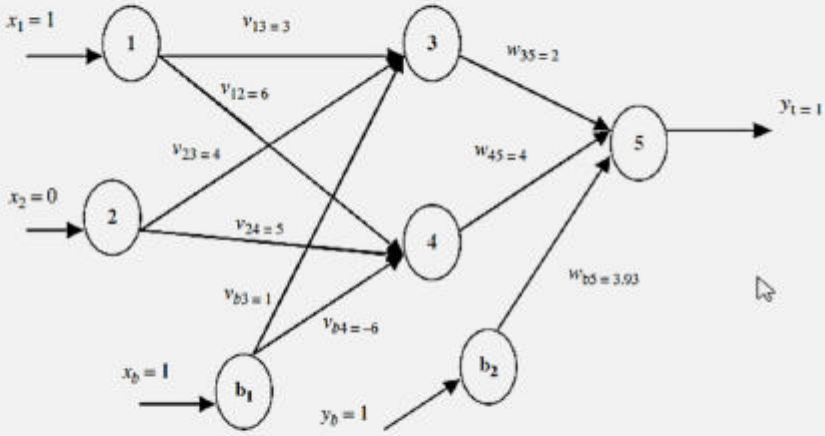
$$v_{14}^{new} = v_{14}^{old} - \eta\beta_{m4}x_1 = 6.0 - (0.1)(-0.1225)(1) = 6.0125$$

$$v_{23}^{new} = v_{23}^{old} - \eta\beta_{m3}x_2 = 4.0 - (0.1)(-0.0043)(0) = 4.0$$

$$v_{24}^{new} = v_{24}^{old} - \eta\beta_{m4}x_2 = 5.0 - (0.1)(-0.1225)(0) = 5.0$$

$$v_{b3}^{new} = v_{b3}^{old} - \eta\beta_{m3}x_b = 1.0 - (0.1)(-0.0043)(1) = 1.004$$

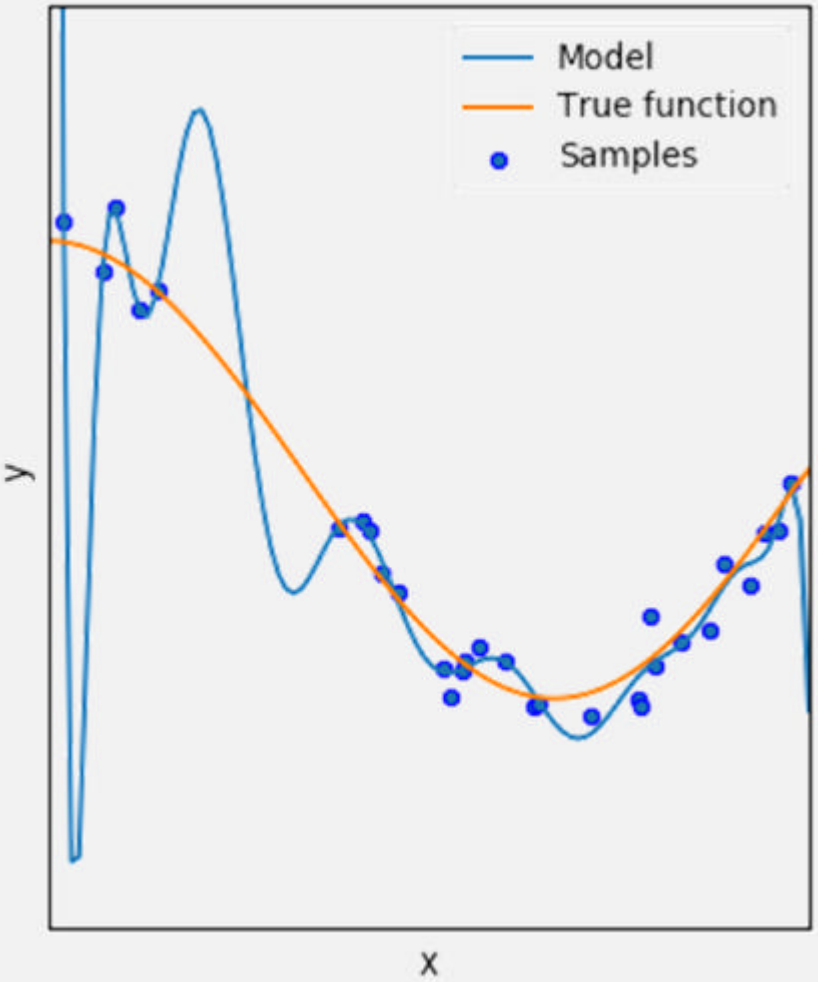
$$v_{14}^{new} = v_{14}^{old} - \eta\beta_{m4}x_1 = -6.0 - (0.1)(-0.1225)(1) = -5.9878$$



With the updated weights, if one were to do second iteration, he would, at the end of the forward pass, obtain the error $(E) = 0.476$. Note that in the first iteration the error was computed as $E = 0.49$. That means as the iteration continues the error decreases.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

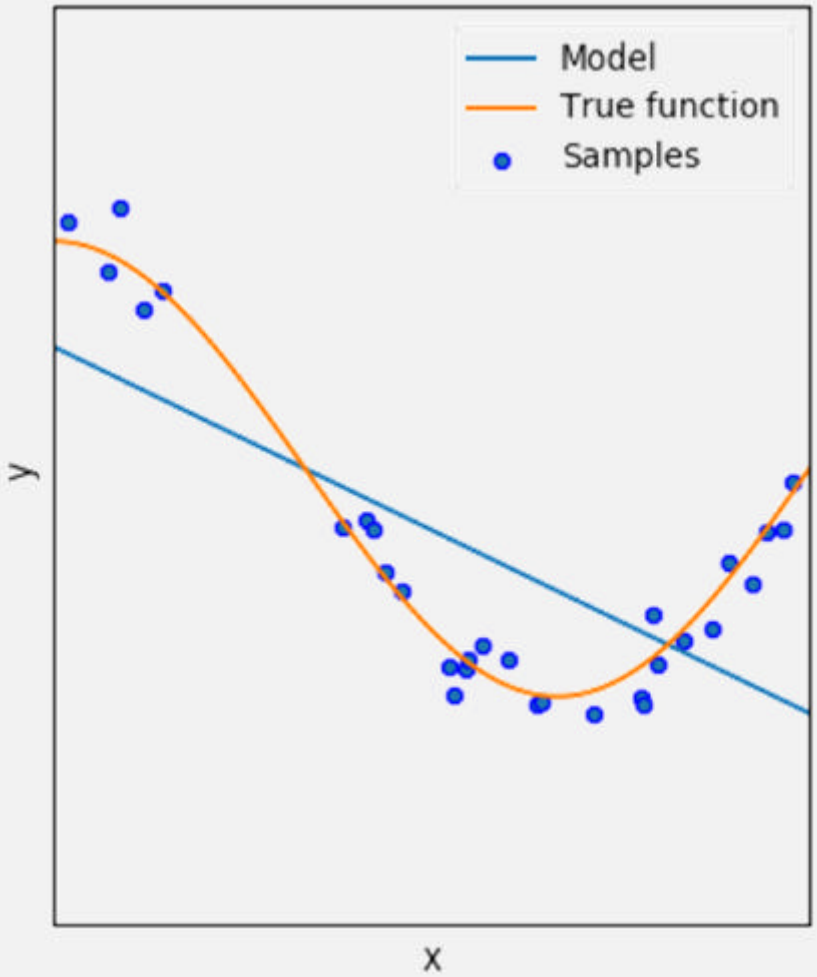
بیش برازش (Overfitting)



- مدل بیش‌برازش، مدلی بسیار پیچیده برای داده‌ها است. به این معنی که در تحلیل رگرسیونی، مدلی با بیشترین پارامترها ایجاد می‌شود.
- در چنین حالتی، مدل با تغییرات جهشی سعی در پوشش داده‌های حاصل از نمونه و حتی مقدارهای نویز می‌کند. در حالیکه چنین مدلی باید منعکس کننده رفتار جامعه باشد. در این گونه موارد، اگر مدل بدست آمده، برای پیش‌بینی نمونه دیگری به کار رود، مقدارهای پیش‌بینی شده اصلا مناسب به نظر نخواهند رسید.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

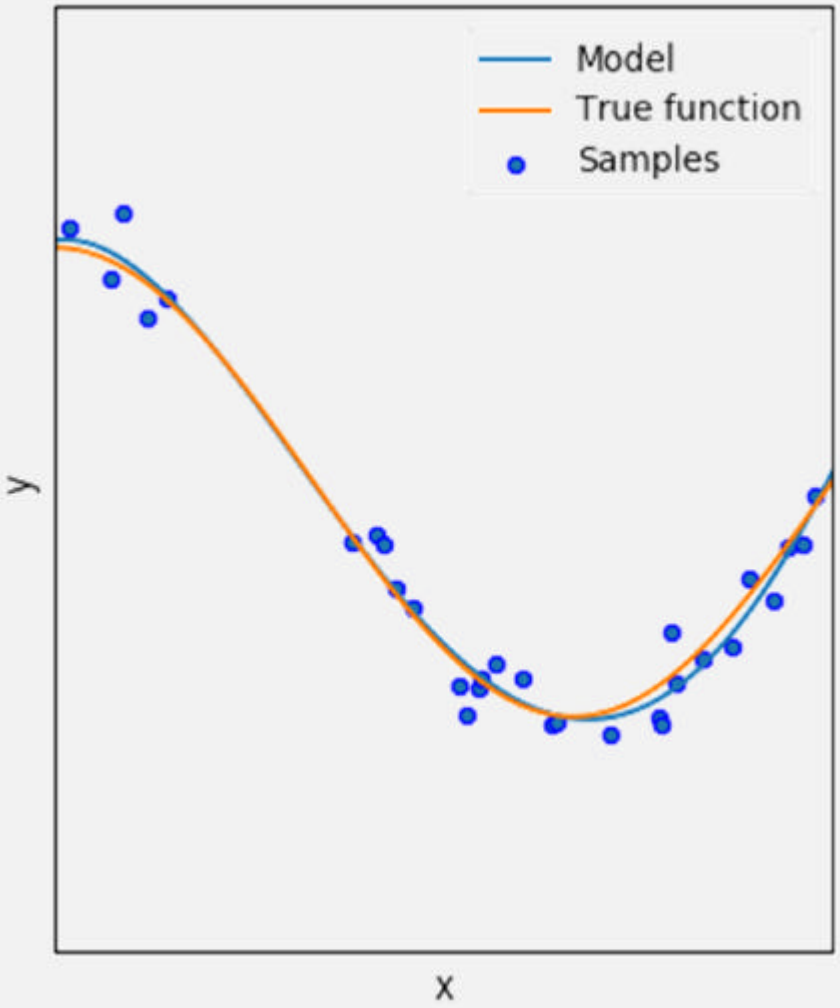
کم برازش (Underfitting)



- همچنین در زمانی که پارامترهای مدل به صورت کم برازش برآورد می‌شوند، جانب احتیاط حفظ شده و مدل سعی می‌کند با کمترین پارامترها، عمل برازش را انجام دهد.
- در نتیجه خطای حاصل از این مدل حتی براساس نمونه‌های به کار رفته نیز بسیار زیاد است. در تصویر ۲، یک نمونه از مدل کم‌برازش دیده می‌شود. درجه منحنی به کار رفته در این حالت ۱ است که معادله خط محسوب می‌شود.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

برازش مناسب



- انتظار ما از یک تحلیل رگرسیون مناسب، ایجاد مدلی است که نه تنها بتواند برای داده‌های مربوط به نمونه برازش مناسب را انجام دهد، بلکه برای داده‌هایی جدید نیز امکان برآورد مناسب وجود داشته باشد.

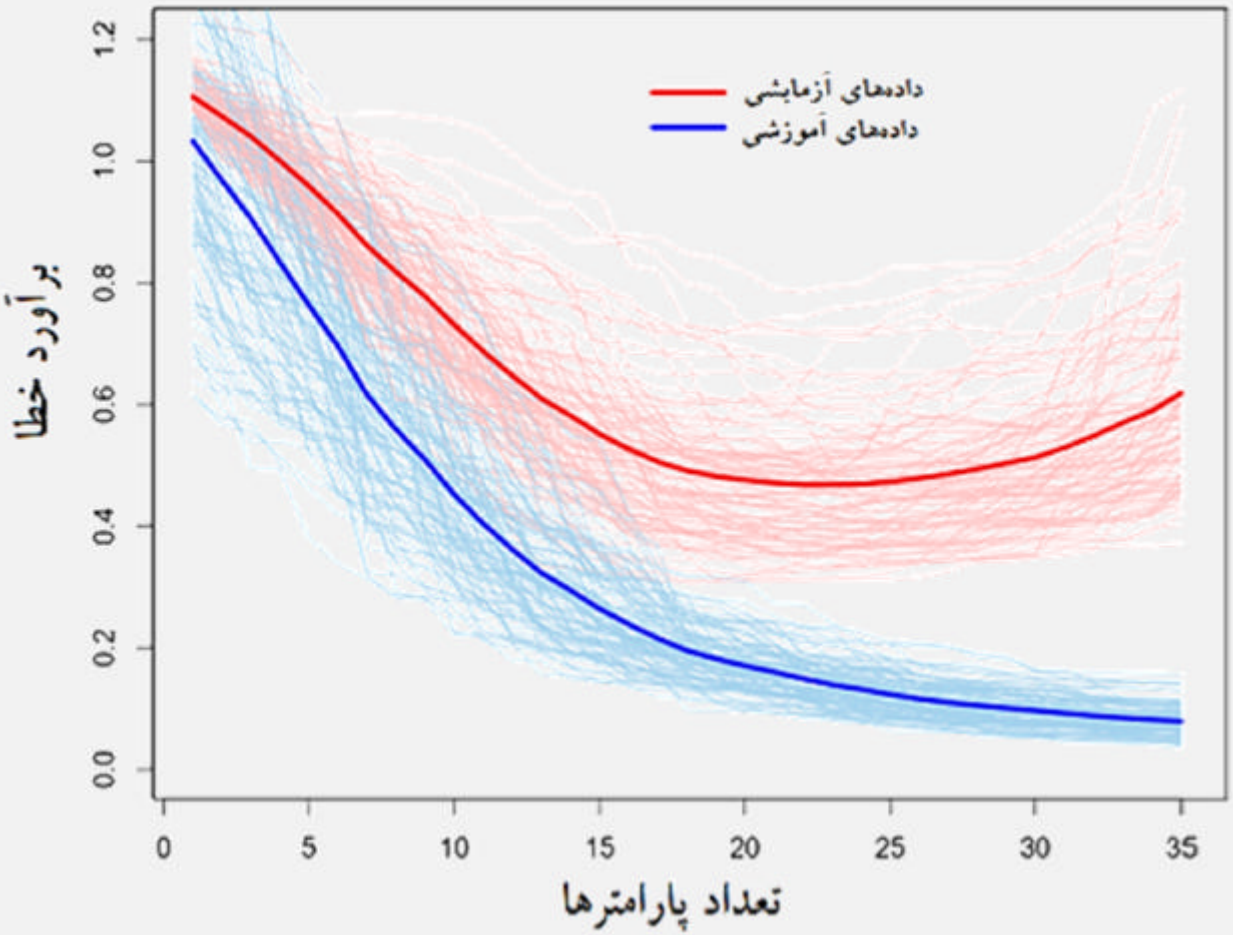
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

روش‌های سنجش اعتبار مدل و جلوگیری از بیش برازش

- **ارزیابی براساس فرضیاتی که باید مدل در آن‌ها صدق کند.**
  - ✓ در این روش تکیه بر داده‌هایی است که مشاهده شده و در ساختن مدل به کار رفته‌اند.
  - ✓ برای مثال در رگرسیون خطی، فرض بر این است که باقی‌مانده‌های مدل رگرسیونی باید تصادفی و با واریانس ثابت باشند. همچنین توزیع آن‌ها نیز باید نرمال باشد.
  - ✓ بررسی صحت این فرضیات می‌تواند به عنوان معیاری برای سنجش اعتبار مدل محسوب شود. همانطور که می‌دانید، پارامترهای مدل رگرسیونی با کمینه سازی مربعات خطا حاصل می‌شود.
  - ✓ بنابراین انتظار داریم که مدل ساخته شده نسبت به هر مدل دیگری کمترین مجموع مربعات خطا را نیز داشته باشد.
- **ارزیابی براساس کارایی مدل در پیش‌بینی مقدارهای جدید (مشاهده نشده).**
  - ✓ از این روش تحت عنوان اعتبار سنجی عرضی (Cross Validation) نام برده می‌شود.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

علل ایجاد بیش برازش در مدلسازی به روش شبکه عصبی مصنوعی



- افزایش تعداد نرون ها در لایه‌ها مخفی
- افزایش تعداد اپوک‌های آموزش
- افزایش تعداد پارامترهای ورودی

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

کنترل بیش برازش

- فرض کنید مشاهداتی از جامعه به صورت یک نمونه تصادفی در دسترس است که قرار است از آن‌ها در مدل‌سازی استفاده شود. هدف در اعتبارسنجی متقابل، دستیابی به مدلی است که تعداد پارامترهای آن بهینه باشد. یعنی پیدا کردن مدلی است که دچار بیش‌برازش نباشد. برای برای دستیابی به این هدف در «آموزش ماشین» معمولا داده‌ها را به دو قسمت تفکیک می‌کنند.
- قسمت داده‌های آموزشی (Training set): از این بخش از داده‌ها به منظور ایجاد مدل و برآورد پارامترهای آن استفاده می‌شود.
- قسمت داده‌های آزمایشی (Testing set): این قسمت از داده‌ها برای بررسی کارایی مدل استفاده می‌شود. اهمیت این بخش از داده‌ها در این نکته است که این مشاهدات شامل مقدارهای متغیرهای مستقل (xها) و پاسخ (y) هستند که در مدل به کار نرفته ولی امکان مقایسه مقدار پیش‌بینی شده را با مقدار واقعی به ما می‌دهند.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

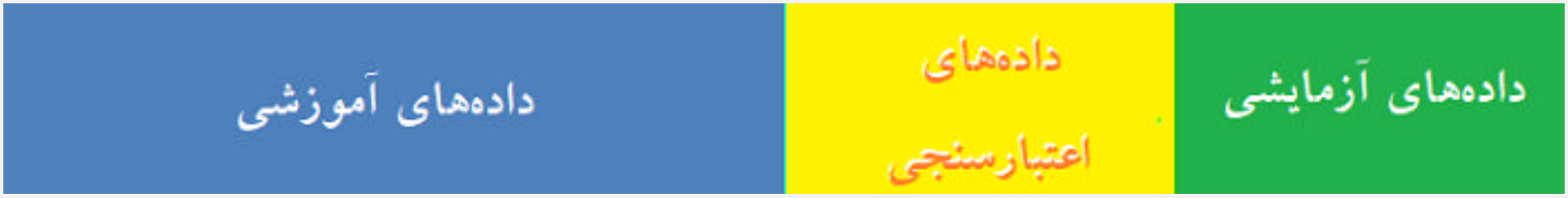
انواع روش‌های اعتبارسنجی عرضی

- روش Holdout
- روش Leave-One-Out
- روش Leave-P-Out
- روش k-Fold

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

روش Holdout

- در این روش به طور تصادفی، داده‌ها به دو بخش آموزشی و اعتبارسنجی تقسیم می‌شود.
- پارامترهای مدل توسط داده‌های آموزشی برآورد شده و برآورد خطای مدل نیز براساس داده‌های اعتبارسنجی محاسبه می‌شود.
- سادگی محاسبات و عدم تکرار فرآیند CV در این روش از مزیت‌های آن محسوب می‌شود. اگر داده‌های مربوط به بخش آموزش و اعتبارسنجی همگن باشند، این روش مناسب به نظر می‌رسد. ولی از آنجایی که محاسبات خطای مدل براساس فقط یک مجموعه داده، بدست آمده ممکن است برآورد مناسبی برای خطای مدل ارائه نشود.



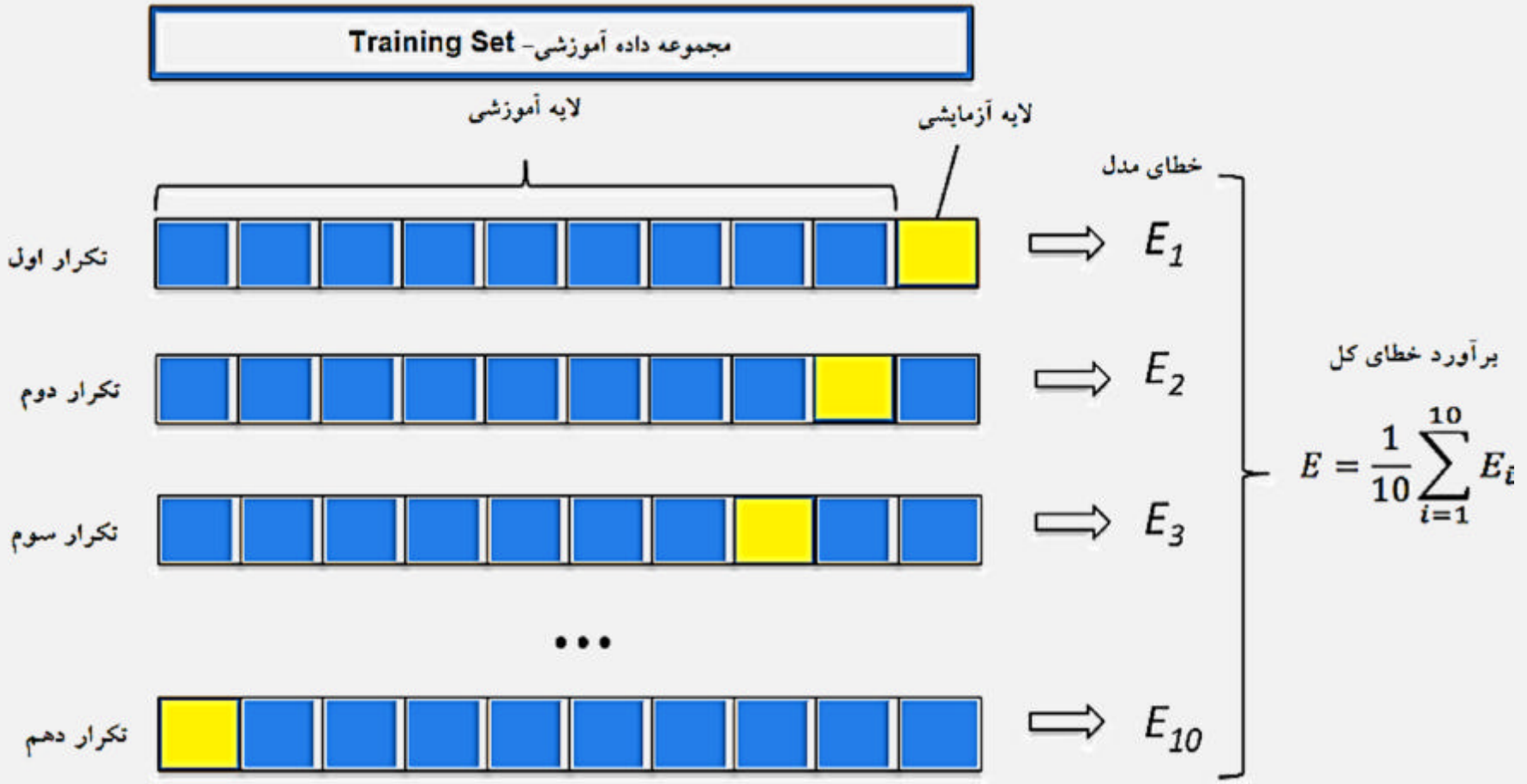
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

روش k-fold

- اگر مجموعه داده‌های آموزشی را به طور تصادفی به k زیرنمونه یا «لایه» (Fold) با حجم یکسان تفکیک کنیم، می‌توان در هر مرحله از فرایند CV، تعداد k-1 از این لایه‌ها را به عنوان مجموعه داده آموزشی و یکی را به عنوان مجموعه داده اعتبارسنجی در نظر گرفت.
- تصویر زیر، مراحل روش k-Fold را به خوبی نشان می‌دهد. مشخص است که با انتخاب k=10، تعداد تکرارهای فرآیند CV برابر با ۱۰ خواهد بود و دستیابی به مدل مناسب به سرعت امکان‌پذیر می‌شود.

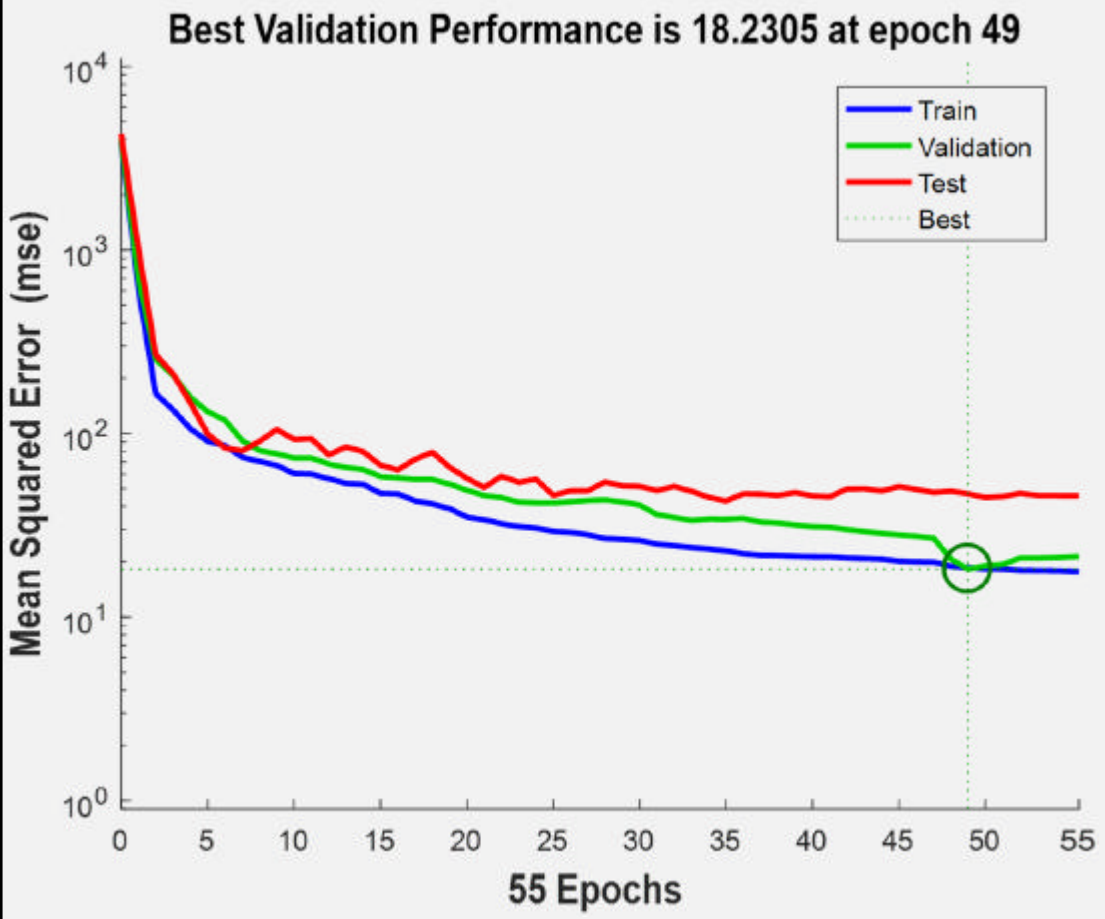
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

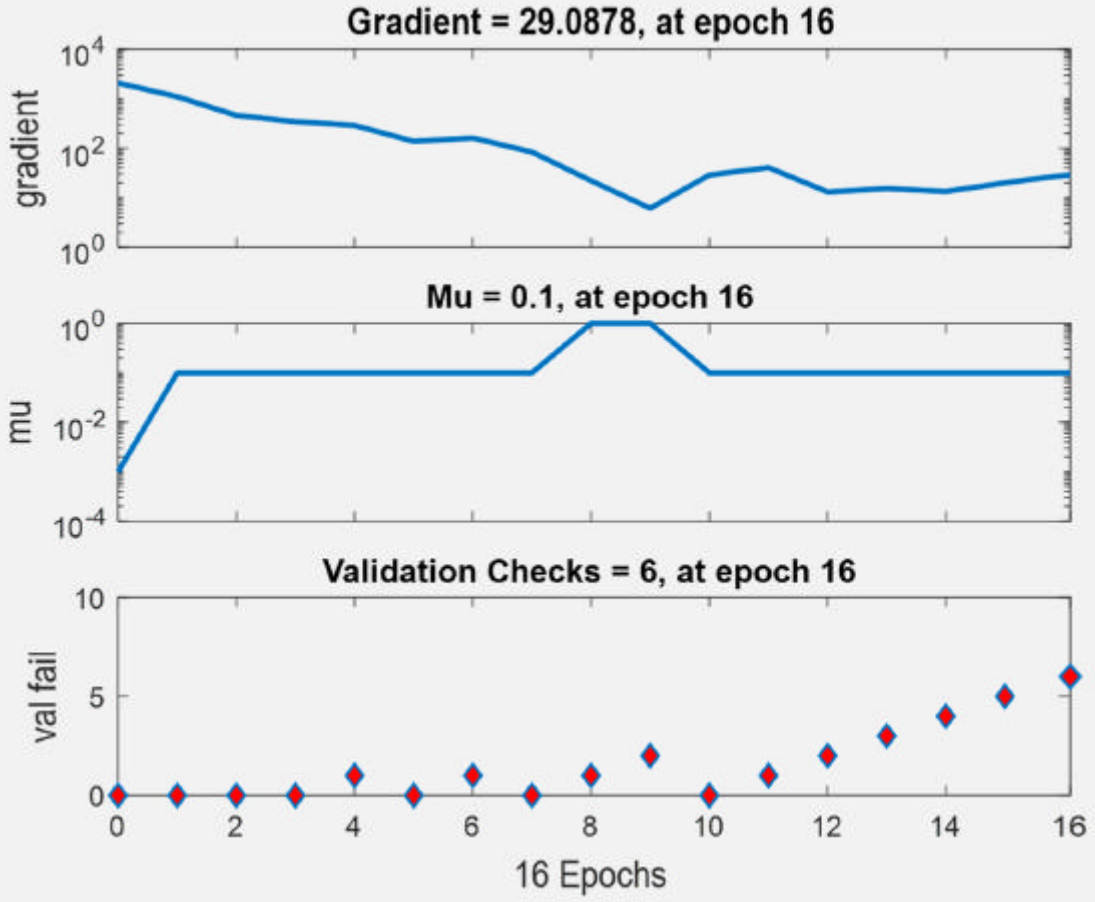
روش k-fold



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

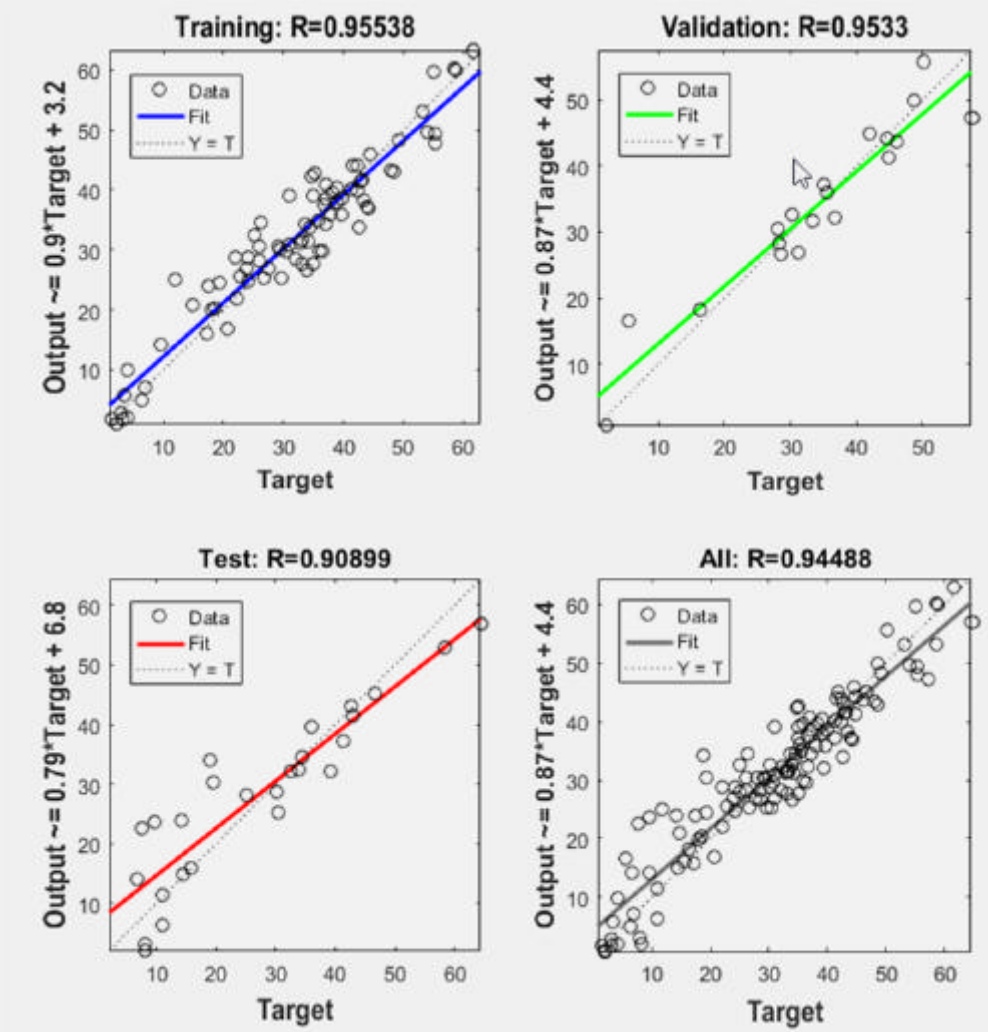
معیارهای توقف آموزش شبکه عصبی مصنوعی

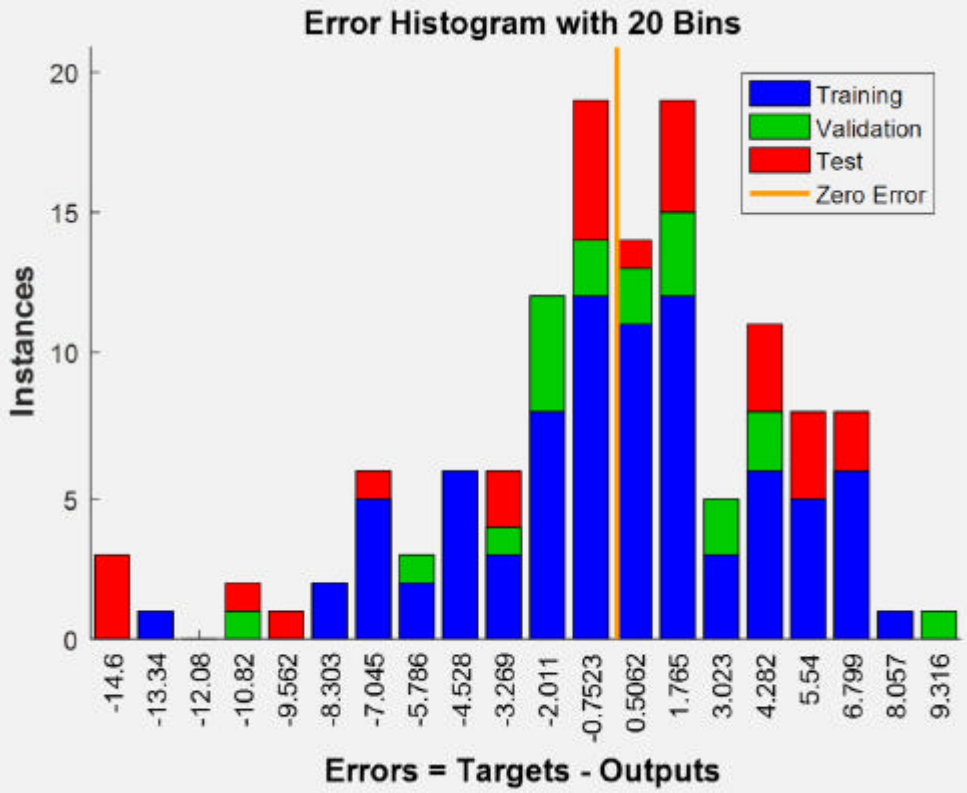




Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

عملکرد و دقت شبکه عصبی مصنوعی





Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

روش k-fold

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(t_i - O_i)^2}$$

$$R^2 = \left[\frac{N\sum_{i=1}^{N} t_i O_i - \sum_{i=1}^{N} t_i \sum_{i=1}^{N} O_i}{\sqrt{N\sum_{i=1}^{N} t_i^2 - \left(\sum_{i=1}^{N} t_i\right)^2}\sqrt{N\sum_{i=1}^{N} O_i^2 - \left(\sum_{i=1}^{N} O_i\right)^2}}\right]^2$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه ها عصبي پايه شعاعي

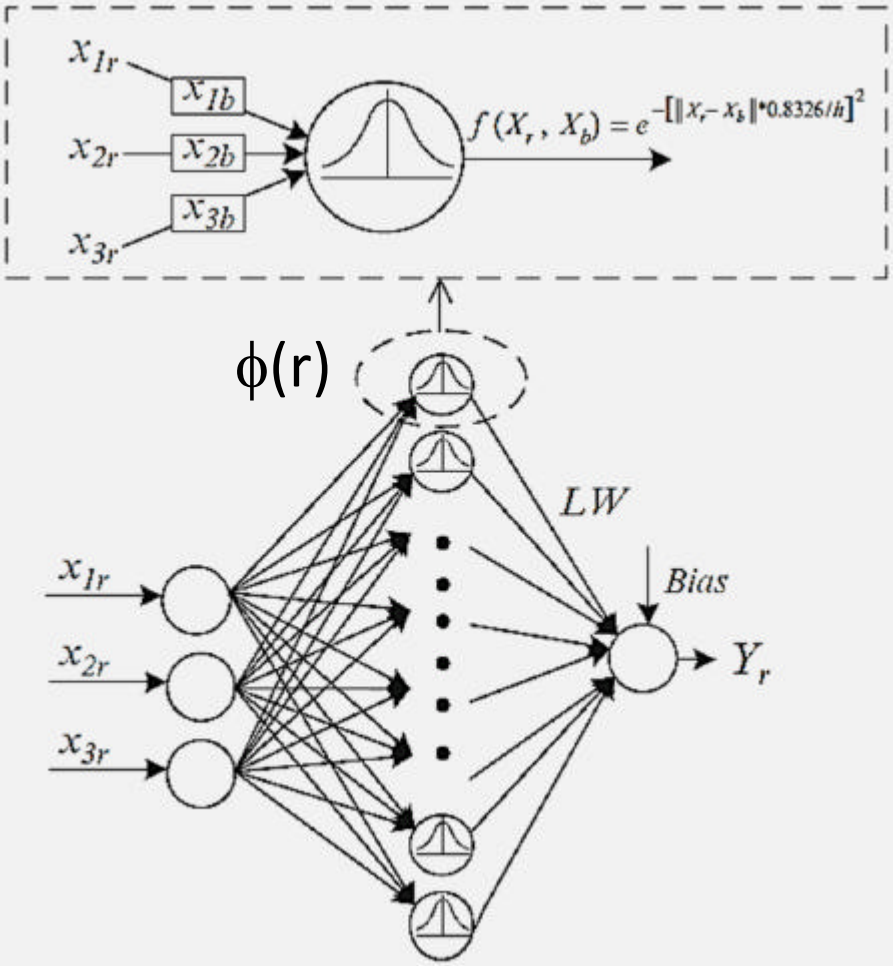
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی پایه شعاعی (RBF)

- شبکه عصبی پایه شعاعی برای اولین بار در سال ۱۹۸۸ توسط (Broomhead & Lowe) پیشنهاد شد.
- دارای یک لایه ورودی، یک لایه پنهان و یک لایه خروجی است.
- در لایه مخفی از توابع پایه شعاعی (RBF) استفاده شده است.
- لایه خروجی دارای تابع انتقال خطی است.
- آموزش این شبکه نسبت به شبکه‌های MLP بسیار سریع‌تر انجام می‌شود.
- شبکه های تابع پایه شعاعی در برخی مسائل طبقه‌بندی و تخمین الگو دارای عملکرد بهتری نسبت به شبکه‌های MLP می‌باشند.

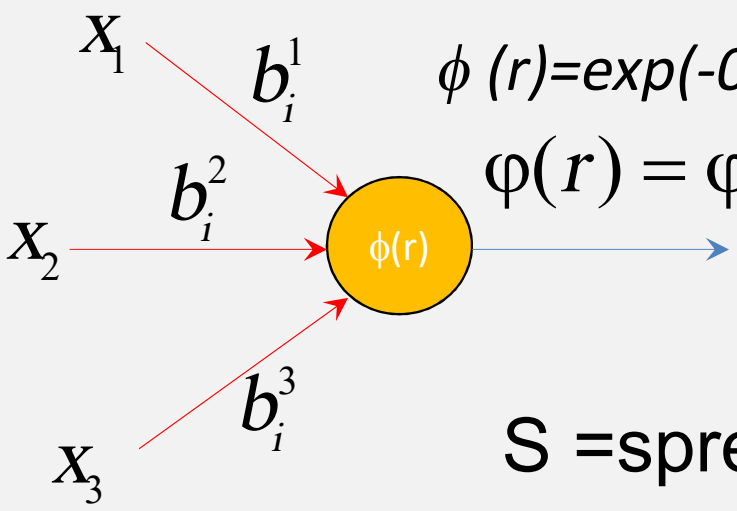
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه تابع پایه شعاعی (Radial Basis Function Network)



Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

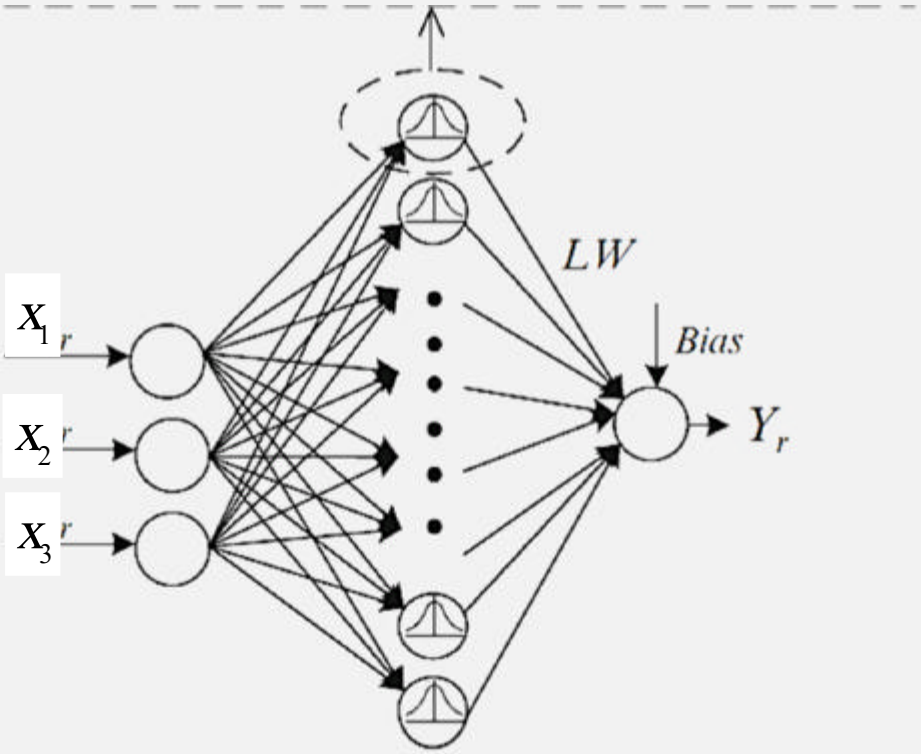
شبکه تابع پایه شعاعی (Radial Basis Function Network)



$\phi(r)=\exp(-0.8326 \times r/s)^2$

$$\varphi(r) = \varphi(\|x - b_i\|)$$

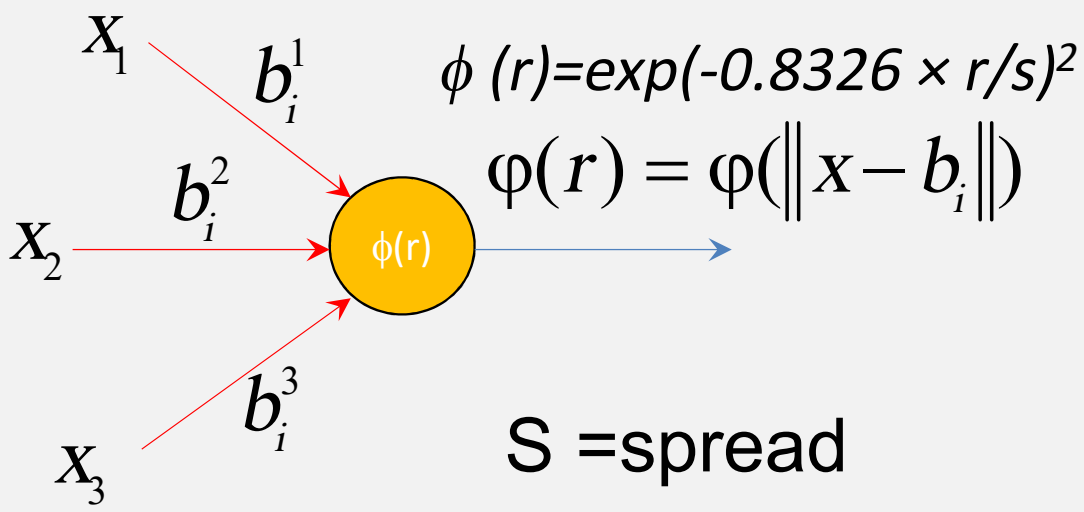
S =spread



$$\|x - y\| = \sqrt{(x-y)^t(x-y)} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه تابع پایه شعاعی (Radial Basis Function Network)



- $x \in R^n$: input vector
- c_i vector value parameter centroid (first layer weight)
- w_i connection weights in the second layer (from hidden layer to output)
- ϕ: activation function should be radially symmetric (i.e. if $\|x_1\| = \|x_2\|$ then $\phi(\|x_1\|) = \phi(\|x_2\|)$)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

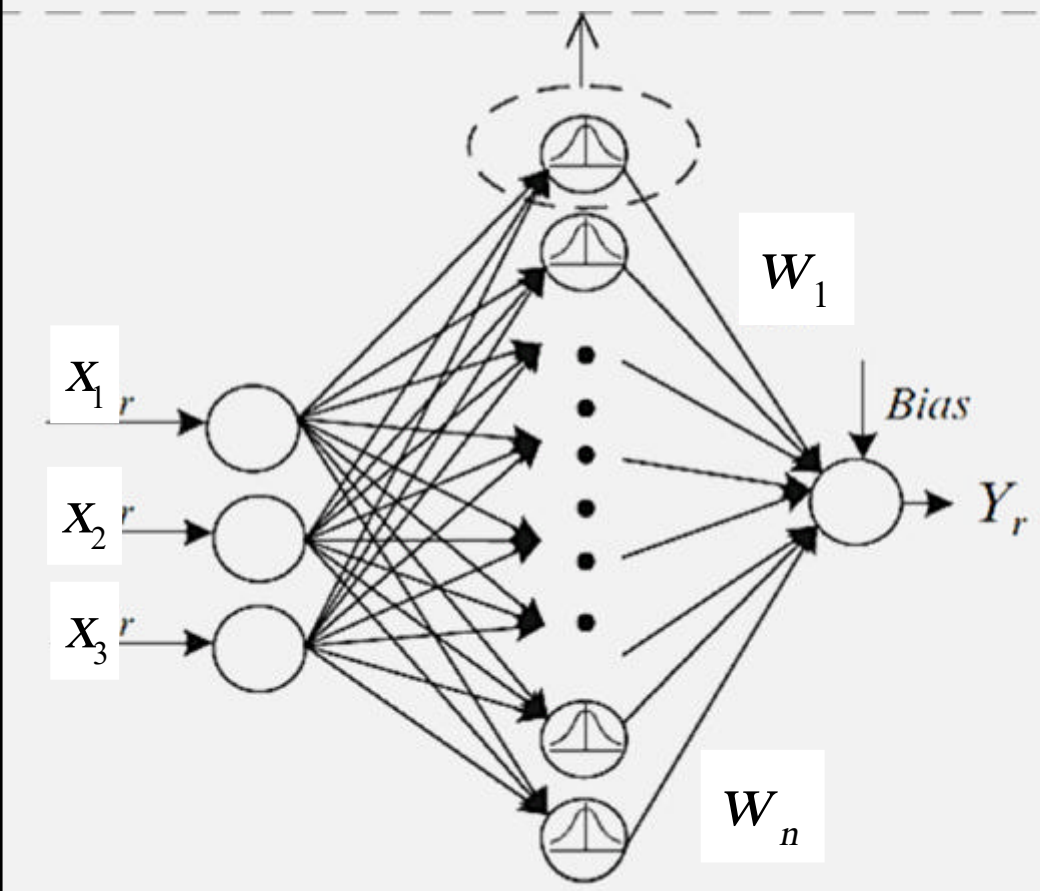
شبکه تابع پایه شعاعی (Radial Basis Function Network)

1. Linear Function: $\phi(r) = r$
2. Cubic Function: $\phi(r) = r^3$
3. Gaussian Function $\phi\,(r)=exp(-0.8326 \times r/s)^2$
4. Multi-Quadratic $\phi(r) = (r^2 + \sigma^2)^{1/2}$
5. Generalized Multi-Quadratic $\phi(r) = (r^2 + \sigma^2)^{\beta}, \quad 1 > \beta > 0$
6. Inverse Multi-Quadratic $\phi(r) = (r^2 + \sigma^2)^{-1/2}$
7. Generalized Inverse Multi-Quadratic $\phi(r) = (r^2 + \sigma^2)^{-\alpha}, \; `\alpha > 0$
8. Thin Plate Spline $\phi(r) = r^2 ln(r)$
9. Shifted Logarithm $log(r^2 + \sigma^2)$

where $r = (\|x - c_i\|)$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

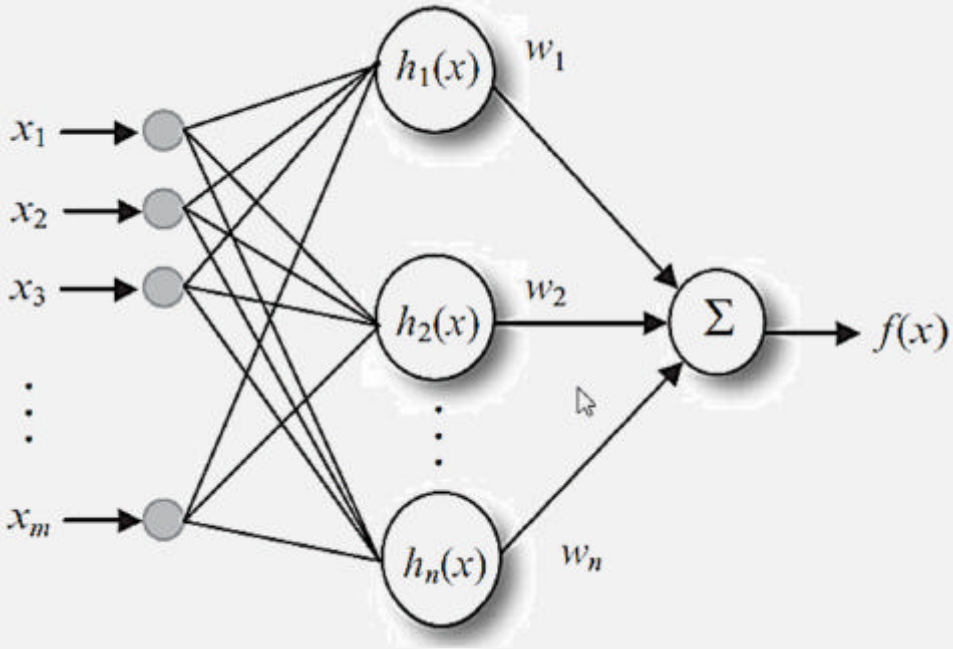
شبکه تابع پایه شعاعی (Radial Basis Function Network)



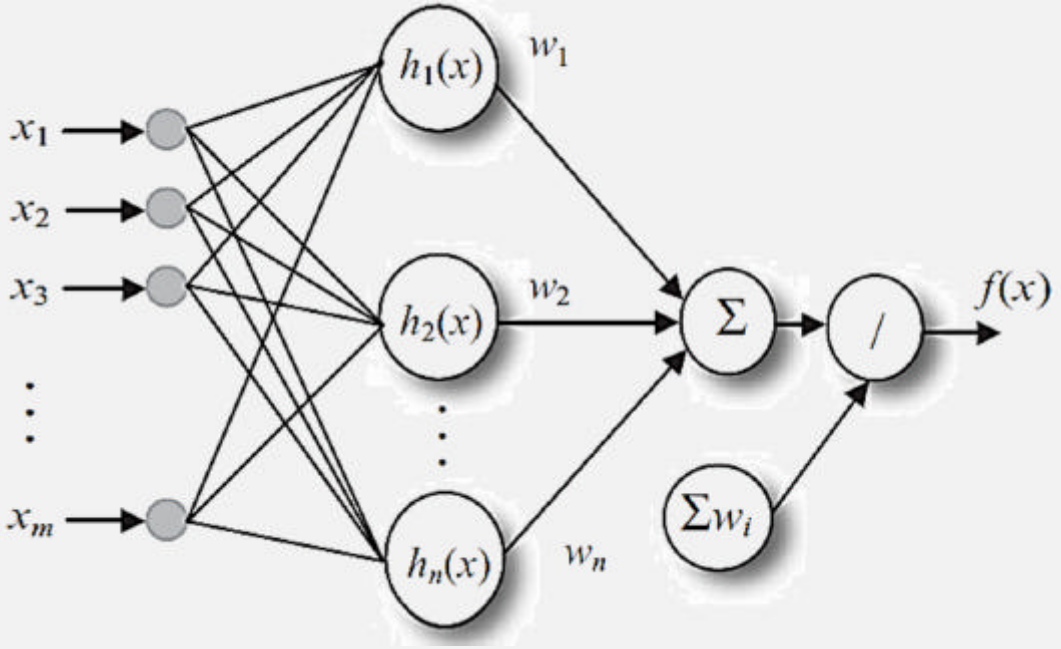
$$Y = f(x) = \sum_{i=1}^{n} w_i \varphi_i(\|x - b_i\|) + Bias$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه تابع پایه شعاعی (Radial Basis Function Network)



Simple RBF network



Weighted average RBF network

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه هاي عصبي رگرسيون عمومي

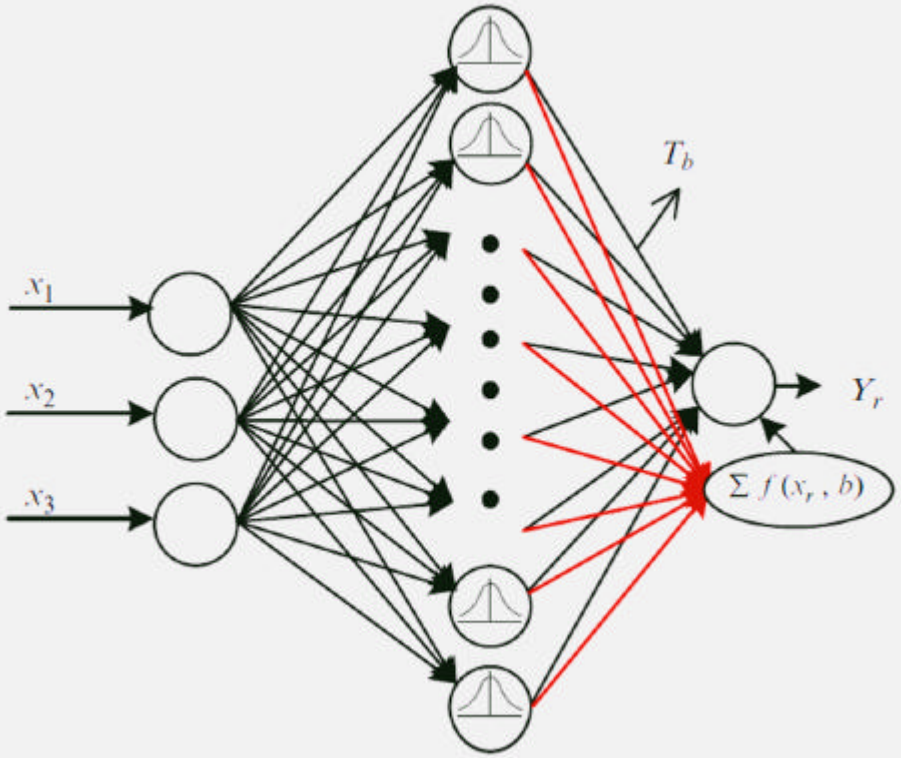
Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی رگرسیون عمومی (GRNN)

- شبکه عصبی رگرسیون عمومی در سال ۱۹۹۱ توسط (D.F. Specht) پیشنهاد شد.
- این شبکه عصبی می‌تواند برای مقاصد رگرسیون و طبقه‌بندی مورد استفاده قرار گیرد.
- مانند شبکه‌های تابه پایه شعاعی (RBF) ، شبکه های GRNN نیاز به آموزش از طریق پس انتشار ندارند و آموزش آنها بسیار سریع انجام می‌شود، به علت استفاده از توابع پایه شعاعی دارای دقت بالایی در پیش‌بینی هستند، می‌توانند نویز در ورودی‌ها را پوشش دهند و برای توسعه آناها نیاز به تعداد زیاد داده نمی باشد.
- در صورتی که تعداد داده های آموزش زیاد باشد کارا نمی‌باشند.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی رگرسیون عمومی (GRNN)



$$Y_r = \frac{1}{\sum_{b=1}^{n} f(X_r, b)} \sum_{b=1}^{n} [f(X_r, b) \times T_b]$$

T_b =target associated with the b^{th} observation

n = number of observations.

$$\varphi(r) = \varphi(\|x - b_i\|) = \exp(-0.8326 \times r/s)^2$$

$$\|x - y\| = \sqrt{(x-y)^t (x-y)} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- تعداد نرون‌ها در لایه مخفی برابر با تعداد داده های مشاهده شده است
- تعداد نرون ها در لایه ورودی و خروجی به ترتیب برابر با تعداد متغیرهای ورودی و تعداد متغیرهای خروجی است.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

مراحل مدلسازي با روش شبكه عصبي مصنوعي

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی پایه شعاعی (RBF)

$$X_i^n = \frac{X_i - X_{\min}}{X_{\max} - X_{\min}}; 0 \le x_i \le 1$$

- **آماده‌سازی پایگاه داده**
  - کنترل داده‌ها و عدم وجود داده‌های تهی
  - نرمال سازی‌داده‌ها
- **انتخاب معماری** (نوع شبکه، تعداد لایه‌های مخفی، تعداد نرون‌ها در لایه‌های مخفی)
- **آموزش شبکه عصبی**
- **شبیه‌سازی**
- **پس پردازش** (ترسیم نمودارهای برابری، کنترل نرمال بودن باقیمانده‌ها، تحلیل حساسیت و تحلیل پارامتریک)

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

نرمال سازی داده‌ها

$$x_i^n = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}; 0 \le x_i \le 1$$

$$x_i^n = 2\frac{x_i - x_{\min}}{x_{\max} - x_{\min}} - 1; -1 \le x_i \le 1$$

$$gain = \frac{2}{x_{\max} - x_{\min}}$$

$$x_i^n = (b-a)\frac{x_i - x_{\min}}{x_{\max} - x_{\min}} + a; a \le x_i \le b$$

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

نرمال سازی داده‌ها

```
%I=Input Matrix (No. of row is number of samples and No. of cols is number of elements
%T=Target Matrix (No. of row is number of samples and No. of cols is number of elements

ra = -1;
rb = 1;
a = min(I(:));
b = max(I(:));
In = (((ra-rb) * (I - a)) / (b - a)) + rb;
a = min(T(:));
b = max(T(:));
Tn = (((ra-rb) * (T - a)) / (b - a)) + rb;
```


Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی پیش خور چند لایه

```
% One Hidden layer
net = newff(P,T, {Hn1},{'tansig' 'purelin'},'trainlm');

% Two Hidden layers
net = newff(P,T, {Hn1 Hn2},{'tansig' 'tansig' 'purelin'},'trainlm');

% Set train parameters
net.trainParam.max_fail=2;
net.trainparam.epochs=500;
net.trainParam.min_grad=1e-10;
net.trainParam.mu_max=1e60;
net.trainParam.goal = 0;
net.divideFcn = 'divideind';
net.divideParam.trainInd =1:round(N*Trainratio/100);
net.divideParam.valInd=(round(N*Trainratio/100)+1):round(m*(Trainratio+Valratio)/100);
net.divideParam.testInd=(1+round(N*(Trainratio+Valratio)/100)):m;
% Train ANN
[net,tr,O,E,Pf,Af]= train(net,inputs,targets);

% Simulate ANN
Y = sim(net,p);
```

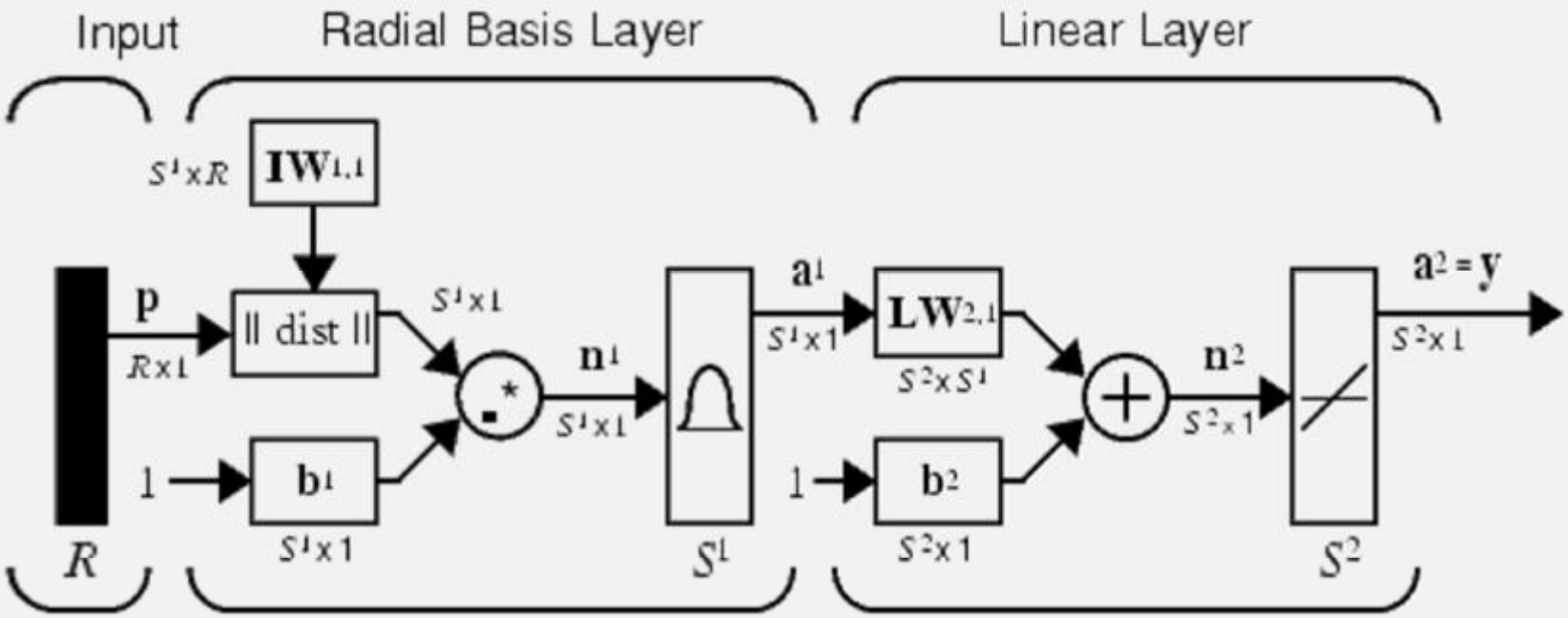

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

الگوریتم های آموزش شبکه عصبی پیش خور چند لایه

Training Function	Algorithm
trainlm	Levenberg-Marquardt
trainbr	Bayesian Regularization
trainbfg	BFGS Quasi-Newton
trainrp	Resilient Backpropagation
trainscg	Scaled Conjugate Gradient
traincgb	Conjugate Gradient with Powell/Beale Restarts
traincgf	Fletcher-Powell Conjugate Gradient
traincgp	Polak-Ribiére Conjugate Gradient
trainoss	One Step Secant
traingdx	Variable Learning Rate Gradient Descent
traingdm	Gradient Descent with Momentum
traingd	Gradient Descent

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی پایه شعاعی (RBF)



Where...

R = number of elements in input vector

S^1 = number of neurons in layer 1

S^2 = number of neurons in layer 2

$a_i^1 = radbas(\| {}_i\mathbf{IW}^{1,1} - \mathbf{p} \| b_i^1)$ $\quad \mathbf{a}^2 = purelin(\mathbf{LW}^{2,1}\,\mathbf{a}^1 + \mathbf{b}^2)$

a_i^1 is i^{th} element of $\mathbf{a}^1$ where ${}_i\mathbf{IW}^{1,1}$ is a vector made of the i^{th} row of $\mathbf{IW}^{1,1}$

The bias b^1 is set to a column vector of 0.8326/SPREAD

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی پایه شعاعی (RBF)

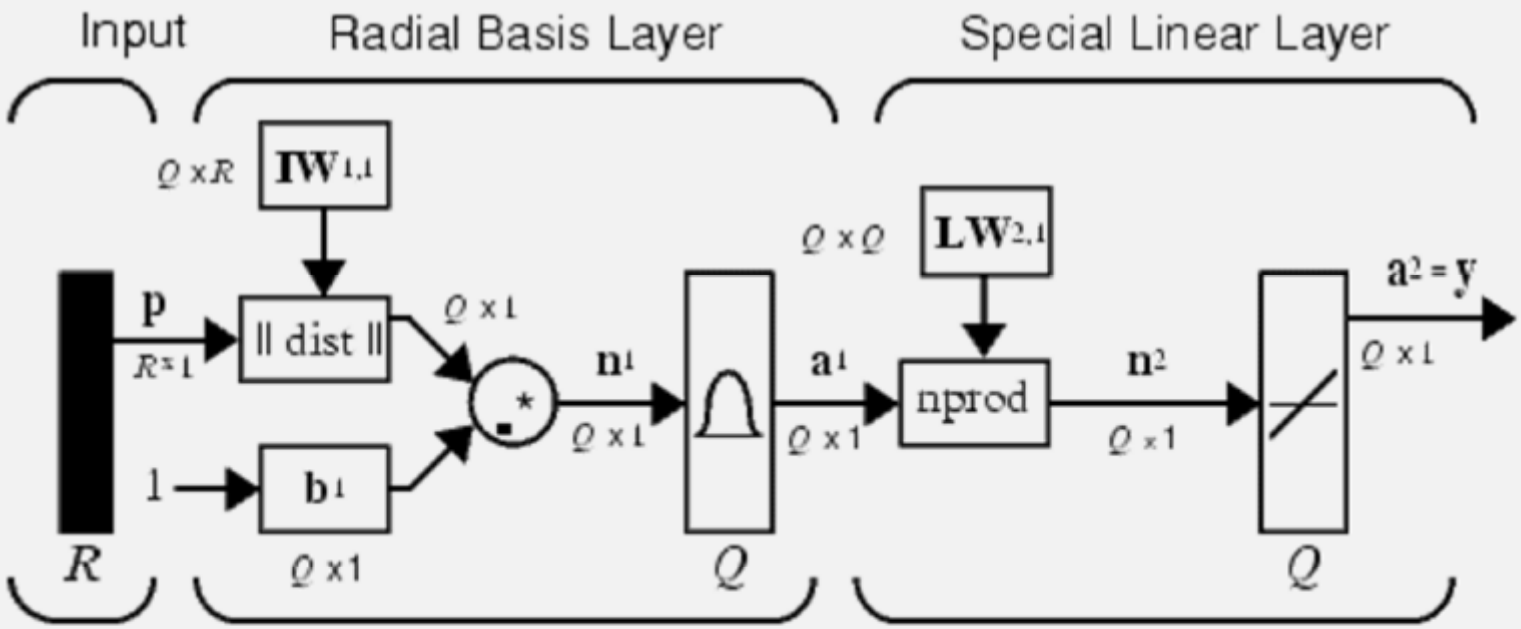
- The function **newrb** iteratively creates a radial basis network one neuron at a time.
- Neurons are added to the network until the sum-squared error falls beneath an error goal or a maximum number of neurons has been reached.
- The call for this function is:

```
net = newrb(P,T,GOAL,SPREAD)
Output=net(P)
```

The function **newrb** takes matrices of input and target vectors, P and T, and design parameters GOAL and, SPREAD, and returns the desired network.

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی رگرسیون عمومی (GRNN)



Where...

R = no. of elements in input vector

Q = no. of neurons in layer 1

Q = no. of neurons in layer 2

Q = no. of input/ target pairs

$$a_i^1 = radbas(\| {}_i\mathbf{IW}^{1,1} - \mathbf{p} \| b_i^1) \qquad \mathbf{a}^2 = purelin(\mathbf{n}^2)$$

a_i^1 is i^{th} element of $\mathbf{a}^1$ where ${}_i\mathbf{IW}^{1,1}$ is a vector made of the i^{th} row of $\mathbf{IW}^{1,1}$

The bias b^1 is set to a column vector of 0.8326/SPREAD

Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology

شبکه‌ها عصبی رگرسیون عمومی (GRNN)

- The call for this function is:

```
net = newgrnn(P,T,spread);
Output=net(P)
```


Ali Reza Ghanizadeh, Ph.D | ghanizadeh@sirjantech.ac.ir
Associate Professor, Department of Civil Engineering, Sirjan University of Technology