Course Description:

This course introduces the fundamental principles and practical aspects of neural networks, focusing on the feedforward neural network, feedback neural network, hybrid intelligent system based on fuzzy-logic systems and genetic algorithm, and applications in modeling, simulation, control, fault diagnosis, information processing, associative memory and optimization computing.

Reference

1. **Zurada Jacek M．Introduction to Artificial Neural System．West Publishing Company, New York, 1992**
2. **D. R. Baughman, Y. A. Liu．Neural Networks in Bioprocessing and Chemical Engineering．Virginia Polytechnic Institute and State University, Blacksburg, 1992**
3. **Nils J. Nilsson(Staford University). Artificial Intelligence: A New Synthesis. Morgan Kaufmann, 1998**
4. **Martin T. Hagan(Oklahoma State University). Neural Network Design. Mark H. Beale(MHB, Inc.) ITP, 1996**
5. **Simon Haykin(McMaster University). Neural Network: A Comprehensive Foundation, 2E. Prentice Hall, 1999**
6. **Jiawei Han, Micheline Kamber. Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000**
7. **袁曾任．人工神经元网络及其应用，清华大学出版社，2000**
8. **孙增圻．智能控制理论与技术，清华大学出版社，1997**
9. **李士勇．模糊控制-神经控制和智能控制论，哈尔滨工业大学出版社，1998**

智能科学与工程

智能科学是探索人类智慧的奥秘与规律及在机器中复现人类的智能，也就是研究模拟人的思维与智能，建立人机结合的系统理论，这是现代科学研究的前沿。

1、研究对智能的产生、形成和工作机制——自然智能理论

2、研究如何用人工的方法模拟、延伸和扩展人的智能，实现某些"机器思维"或脑力活动自动化——人工智能理论

智能工程的任务是构建各种实用的智能系统，研制各种智能系统的开发工具。

智能科学与工程——人-机智能系统。重点在于人的智能与计算机的高性能两者结合，构建人机结合的智能系统。

人工智能研究途径和方法

1、结构模拟，神经计算

根据人脑所具有的生理结构和工作原理，实现计算机的智能。用神经计算的方法实现学习、联想、识别和推理等功能，模拟人脑的智能行为，使计算机表现出某种智能。

2、功能模拟，符号推演

以人脑的心理模型将问题或知识表示成某种逻辑网络、采用符号推演的方法，实现搜索、推理、学习等功能，从宏观上模拟人脑的智能行为，实现机器智能。

3、行为模拟，控制进化

模拟人在控制过程中的智能活动和行为特性，如自寻优、自适应、自学习、自组织等，研究和实现人工智能。

Introduction to Intelligent System

Intelligent System
- Bio-Intelligence
- Machine Intelligence
  - Artificial Intelligence — Expert System
  - Computation Intelligence
    - Artificial Neural Network
    - Fuzzy System
    - Evolutionary Computation
      - Genetic Algorithm
      - Evolutionary Strategy
      - Evolutionary Programming
    - **Rough Set Extension Theory**

A．Artificial Intelligent

Definition of AI

Artificial intelligence is the part of computer science concerned with designing intelligent computer systems, that is, systems that exhibit characteristics we associate with intelligence in human behavior.

The goal of AI from the definition

to make computers "think",

to make computers solve problems requiring human intelligence.

Another definition of AI

Artificial intelligence is the branch of computer science dealing with symbolic, non-algorithm methods of problem solving.

Two aspects of AI-based methods for problem-solving

1. AI does not use an algorithm.

2. AI involves symbolic processing.

major AI-based technologies New

Expert system

Neural networks

Fuzzy-logic systems

Genetic Algorithms

Rough Set

Extension Theory

B. Expert System, Neural Networks, and Fuzzy-Logic System

A expert system is a computer program that uses high-quality, in-depth, knowledge to solve complex and advanced problems typically requiring human experts.

Expert systems operate *symbolically*, on a *macroscopic* scale, processing non-numerical symbols and names.

A neural network is a computing system made up of a number of simple, highly interconnected nodes or processing elements, which process information by its dynamic state response to external inputs.

The goal of a neural network is to map a set of input patterns onto a corresponding set of output patterns.

Neural networks use *subsymbolic processing*, characterized by microscopic interaction that eventually manifest themselves as macroscopic, symbolic, intelligent behavior.

Fuzzy-Logic allows us to mesh a quantitative approach with the qualitative representation. It provides a way to quantify certain qualifiers such as *approximately, often, rarely, several, few, and very*.

To use fuzzy logic, we first need a *fuzzy set*. In a fuzzy set, the transition from membership to non-membership is not well-defined. We quantify the *degree of membership* with values between 0(not a member) and 1(definitely a member).

Expert Network

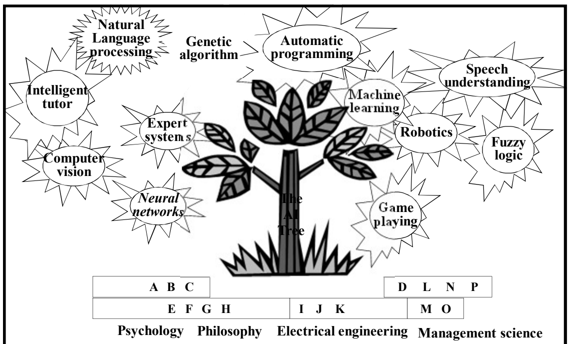
Fuzzy Network

Neural-Fuzzy Networks for Expert Systems

neural network → (Numeric values) → **fuzzifier** → (Fuzzy values) → **expert system** → (Fuzzy values) → **defuzzifier** → (Numeric values)

AI Methods	**AI Features or Functions**
Heuristic Programming	Self – Adaptation
Expert System	Self – Learning
Knowledge Engineering	Self –Recognition
Pattern Recognition	Self – Stabilization
Natural Language Understanding	Self – Turning
Theorem Providing	Self – Coordination
Machine Learning	Self – Organization
Artificial Neural Network	Self – Diagnosis
Fuzzy Logic	Self – Repairing
Intelligent Robot	Self – Reproduction

Rough Set **Extension Engineering**

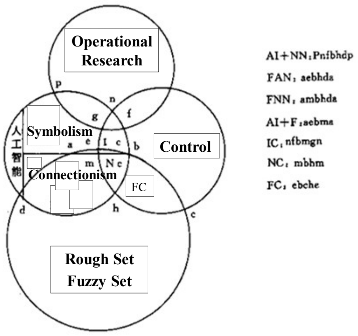


A Psycholinguistics
B Sociolinguistics
C Computational linguistics
D Adaptive systems
E Cognitive psychology
F Philosophy
G Philosophy of language
H Logic
I Robotics
J Image processing
K Pattern recognition
L Management
M Mathematics/statistics
N Operations research
O Management information System
P Biology

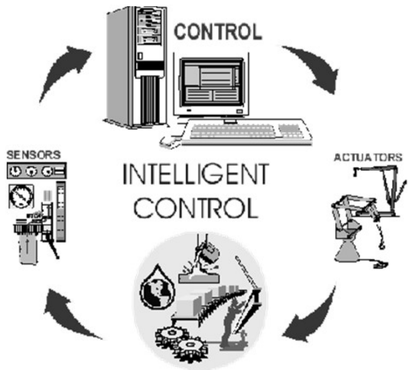
Intelligent Applications

Intelligent Control	**Intelligent Diagnosis**
Intelligent Regulation	**Intelligent Dispatch**
Intelligent Management	**Intelligent Operation**
Intelligent Decision-Making	**Intelligent Software**
Intelligent Instrument	**Intelligent Robot**
Intelligent Machine	**Intelligent Automation**
Intelligent Communication	**Intelligent Computer**
Intelligent Network	**Intelligent Database**
Intelligent Interface	**Intelligent Agent**
Intelligent Monitor	**Intelligent Housing Colony**

Intelligent Control



Intelligent Control



Chapter 1 Introduction

1.1 ANN Development History

1.2 Basic Principle of ANN

1.3 Properties of Neural Networks

1.4 Potential Applications of Neural Networks

1.1 ANN Development History

1943 MP model (McCulloch and Pitts)

1944 Hebb learning rule $\Delta W_{ij}=aS_iS_j \quad a>0$

1957 Perceptron (Rosenblatt)

1962 Adaline (Adaptive linear element) (Widrow)

1969 Book "Perceptron" (Minsky and Papert)

1982 Hopfield network

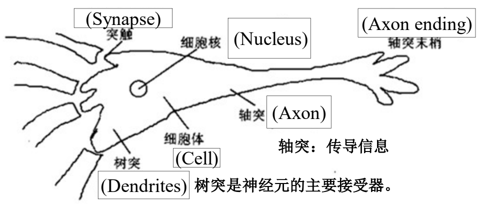
1988 First China neural network conference in Beijing

1992 IEEE neural network conference in China

1.2 Basic Principle of ANN

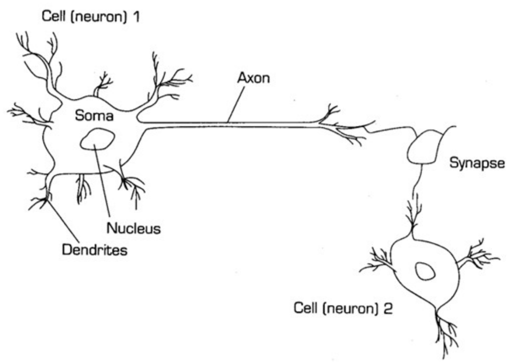
A. Bio-neuron Model 脑神经元由细胞体、树突和轴突构成。

神经元之间通过轴突末梢（输出）与树突（输入）相互联结,其接口称为突触.

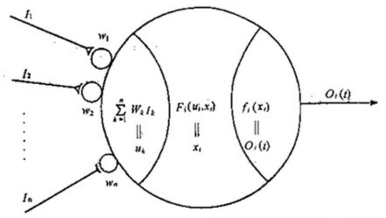


轴突：传导信息

树突是神经元的主要接受器。

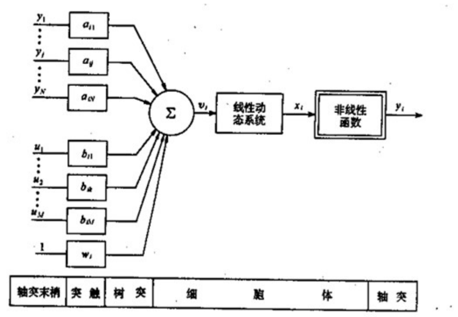


B. Neuron Model

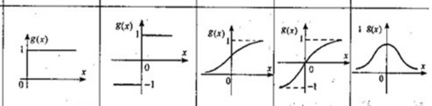


Input	Input processing	Activity processing	Output processing	output

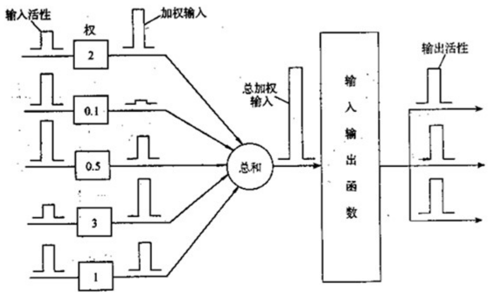
C. Neuron Model Structure



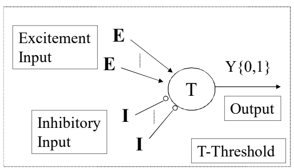
D. Nonlinear Function in a Neuron Model

名称	阈值函数	双向阈值函数	S型函数	双曲正切函数	高斯函数
公式 $g(x)$	$g(x)=\begin{cases}1 & x>0\\ 0 & x\leqslant 0\end{cases}$	$g(x)=\begin{cases}+1 & x>0\\ -1 & x\leqslant 0\end{cases}$	$g(x)=\frac{1}{1+e^{-x}}$	$g(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}}$	$g(x)=e^{-(x^2/\sigma^2)}$
图形					
特征	不可微,类阶跃,正值	不可微,类阶跃,零均值	可微,类阶跃,正值	可微,类阶跃,零均值	可微,类脉冲

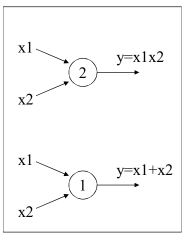
E. Idea Pattern



F. MP Model

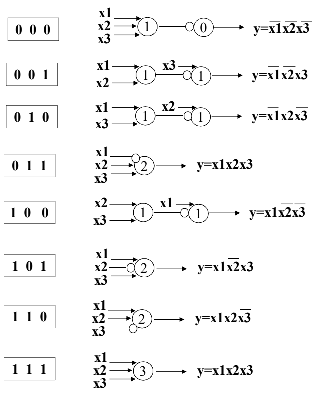


输入条件	输出
∑E≥T, ∑I=0	Y=1
∑E≥T, ∑I>0	Y=0
∑E<T, ∑I=0	Y=0
∑E<T, ∑I≤0	Y=0



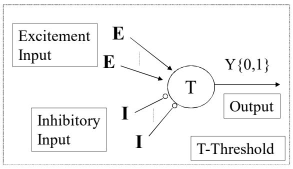
Binary Boolean logic operation by using MP model

Ternary Boole logic operation by using MP model



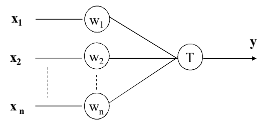
Modified M-P Model:

linear weight neural model



Input Condition	Output
∑E - ∑I≥T	Y=1
∑E - ∑I<T	Y=0

Typical MP Model



y=0, If $\sum_{n=1}^{N} w_n x_n < T$

y=1, If $\sum_{n=1}^{N} w_n x_n \geq T$

$$y = \mathrm{sgn}(\sum_{n=1}^{N} w_n x_n - T)$$

$$\mathrm{sgn}(Z) = \begin{cases} 1, Z \geq 0 \\ -1, Z < 0 \end{cases}$$

1.3 Properties of Neural Networks

A. Strengths of Neural Networks

(1) Information is distributed over a field of nodes.

(2) Neural networks have the ability to learn.

(3) Neural networks allow extensive knowledge indexing.

(4) Neural networks are better suited for processing noisy, incomplete, or inconsistent data.

(5) Neural networks mimic human learning processes.

(6) Automated abstraction.

(7) Potential for online use.

B. Comparison of Neural Networks to Empirical Modeling

First, neural networks have a better *filtering* capacity than empirical models because of the *micro-feature* concept.

Second, neural network are more adaptive than empirical models because of having specified *training algorithms.*

Third, neural networks are truly MIMO systems.

C. Limitations of Neural Networks

(1) Long training times

(2) Large amount of training data

(3) No guarantee of optimal results

(4) No guarantee of 100% reliability

1.4 Potential Applications of Neural Networks

(1) Classification

(2) Predication and Optimization

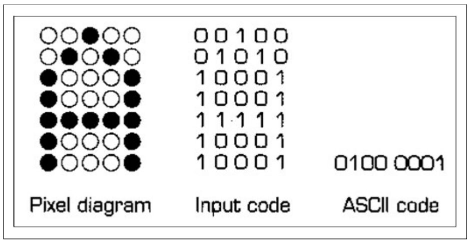
(3) Process - Forecasting, Monitoring, Diagnosis, Modeling and Control

(4) Data - Filtering

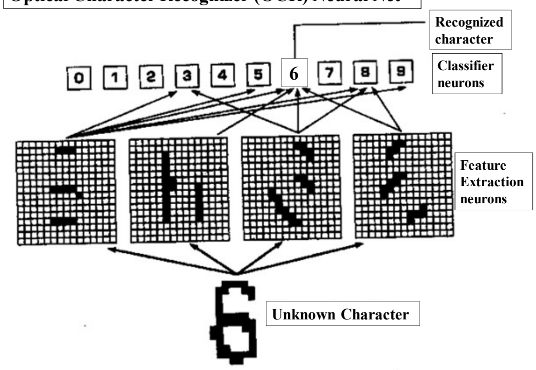
(5) Expert System

• • • • • •

Processing Information in the Network



Optical Character Recognizer (OCR) Neural Net

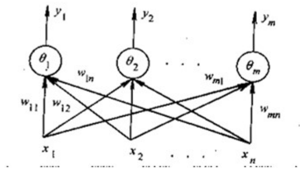


Chapter 2 Perceptron Network

2.1 Single-layer Perceptron Network

2.2 Multi-layers Perceptron Network

2.1 Single-layer Perceptron Network

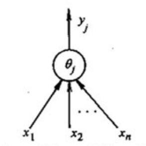


$$x=[x_1,x_2,\cdots,x_n]^T$$ **Input Vector**

w_{ji} weight factor from x_i to y_j

y_j (j=1,2,…,m) output

A. A Neuron



$$S_j=\sum_{i=1}^{n} w_{ji}x_i-\theta_j$$

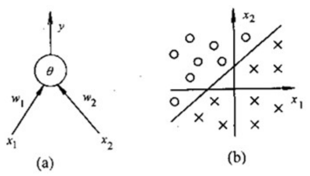
$$y_j=f(s_j)=\begin{cases}1 & s_j\ge 0\\ -1 & s_j\langle 0\end{cases}$$

X^p (p=1,2,….,p) ,

P point in the n dimension space

Perceptron Separates P point into two classes

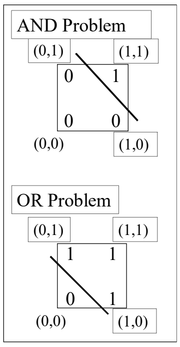
B. Two Dimension Space



Neuron Sample space

Linear equation

$w_1x_1+w_2x_2-\theta=0$



C. Learning Algorithm

1. Initial weight $w_i(0)$, k=0, select at random
2. Select a group of sample, x_p and d_p(desired output)

   Calculate:
   $$s=\sum_{i=0}^{n} w_i x_{pi}$$
   $$y_p=f(s)=\begin{cases}1 & s\geq 0\\ -1 & s\langle 0\end{cases}$$

   Assume: $x_{p0}=1$, $w_0=-\theta$

3. Adjusting weight

   $w_i(k+1)=w_i(k)+\eta(d_p-y_p)\,x_{pi}$ i=1,2,…,n

   η Learning rate

4. Select next sample, and k=k+1

   If $w_i(k+1)=w_i(k)$ Then

   end

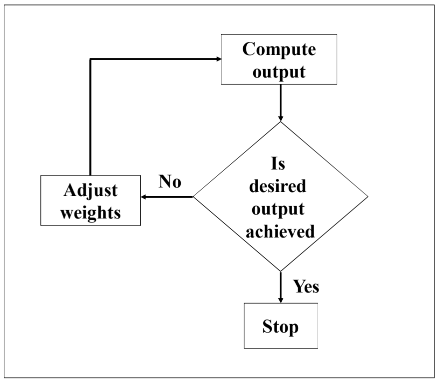
   Else

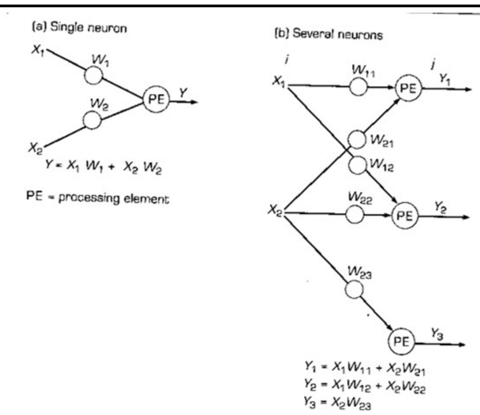
   goto 2

   End if

   i=1,2,…,n

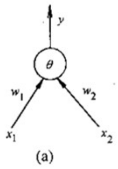
Learning Process of an Artificial Neural Network

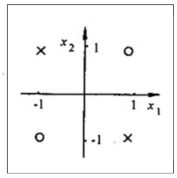




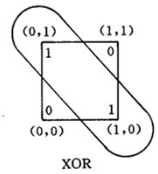
D. Limitation of Simple Perceptron

XOR relation is linear impartibility

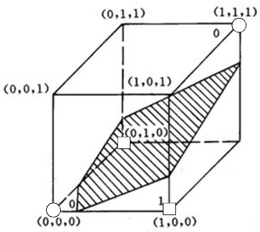




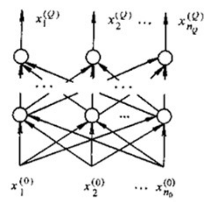
x1	x2	y	
0	0	0	$0\times w1+0\times w2<\theta\ \rightarrow 0<\theta$
0	1	1	$0\times w1+1\times w2>\theta\ \rightarrow w2>\theta$
1	0	1	$1\times w1+0\times w2>\theta\ \rightarrow w1>\theta$
1	1	0	$1\times w1+1\times w2<\theta\ \rightarrow w1+w2<\theta$



Input pattern	Output pattern
0 0 0	0
0 1 0	1
1 0 0	1
1 1 1	0



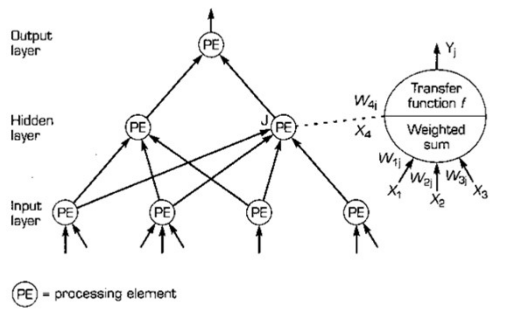
2.2 Multi-layers Perceptron Network



$$s_i^{(q)}=\sum_{j=0}^{n_{q-1}} w_{ij}^{(q)} x_j^{(q-1)}$$
$$x_i^{(q)}=f(s_i^{(q)})=\begin{cases}1 & s_i^{(q)}\geq 0\\ -1 & s_i^{(q)}\langle 0\end{cases}$$

$(x_0^{(q-1)}=\theta_i^{(q)}, w_{i0}^{(q)}=-1)$

i=1,2,…,n_q j=1,2,…,n_{q-1} q=1,2,…,q



XOR Problem

1.design $w_{11}^{(1)}$, $w_{12}^{(1)}$ to get line L_1

L_1 equation:

$w_{11}^{(1)} x_1^{(0)}+w_{12}^{(1)} x_2^{(0)} - \theta_1^{(1)}=0$

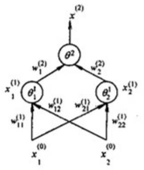
p2 1, p1,p3,p4 -1

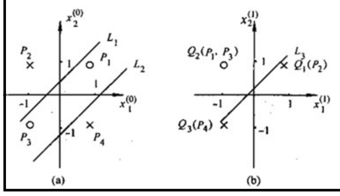
2. design $w_{21}^{(1)}$, $w_{22}^{(1)}$ to get line L_2

p4 -1 p1, p2, p3 1

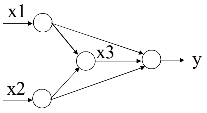
3. There are 3 point q1,q2,q3 in Fig. (b)

Design $w_1^{(2)}$, $w_2^{(2)}$ to get line L_3

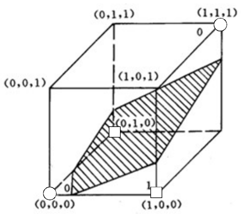




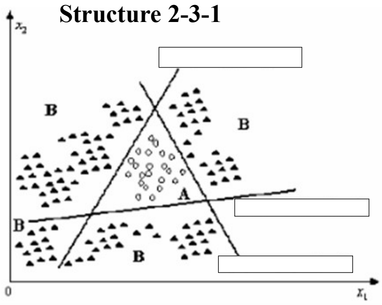
XOR Solution 2



Input pattern	Output pattern
0 0 0	0
0 1 0	1
1 0 0	1
1 1 1	0



Structure 2-3-1



Chapter Highlights

1. The goal of artificial intelligence is to make computers "think" and to make computers solve problems requiring human intelligence.
2. The goal of a neural network is to map a set of input patterns onto a corresponding set of output patterns.
3. An artificial neural network can be organized in many different ways, but the major elements are the processing elements, the connections among the processing elements,the inputs,the outputs,and the weights.
4. Weights express the relative strength (or importance) given to input data.
5. An activation value is translated to an output by going through a transfer function. The output can be related in a linear or nonlinear manner or via a threshold value.
6. ANNs lend themselves to parallel processing. However, most current ANNs are solved on standard computers where multiprocessing is simulated on a single processor (such as simulated ANNs).

Questions for Review

1. What is an artificial neural network?
2. How do weights function in an artificial neural network?
3. Describe the role of the transfer function.
4. List the major benefits of neural computing.
5. List the major limitations of neural computing.

Questions for Discussion

1. Compare neural computing and conventional computing.
2. Explain the combined effects of the summation and transformation functions.
3. Discuss the relationship between a transfer function and a threshold value.
4. Discuss the major advantages of ANNs.
5. What related applications can you think of for using neural computing
6. Explain the difference between MP model and Perceptron.

Chapter 3 Topology and Learning Algorithm of NN

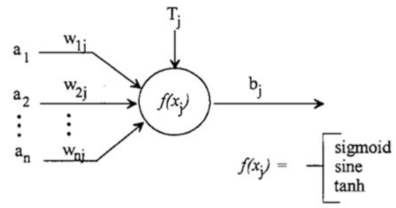
3.1 Components of a Node

3.2 Topology of a Neural Network

3.3 Introduction to learning and Training with NNs

3.1 Components of a Node

A. Anatomy of j^{th} Node



Total Activation $\quad x_j = \Sigma w_{ij} a_i - T_j$

B. Transfer Functions

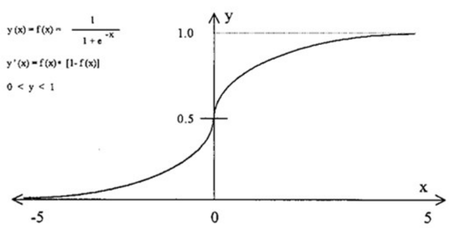
The complete node calculation is :

$$f(x_j) = f(\Sigma (w_{ij} a_i) - T_j)$$

What function form do we choose for f( ) ?

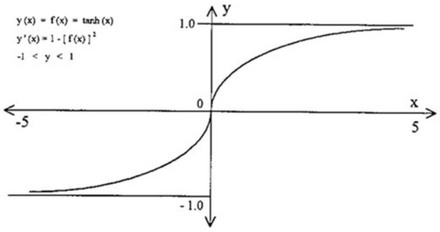
A sigmoid (S-shaped) function

$$f(x) = \frac{1}{1 + e^{-x}} \qquad (2.3)$$



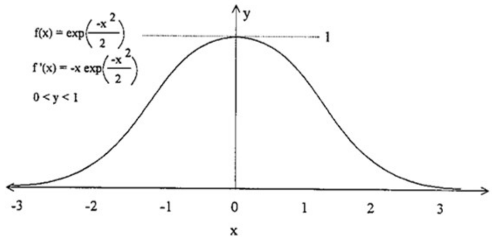
A hyperbolic tangent transfer function

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

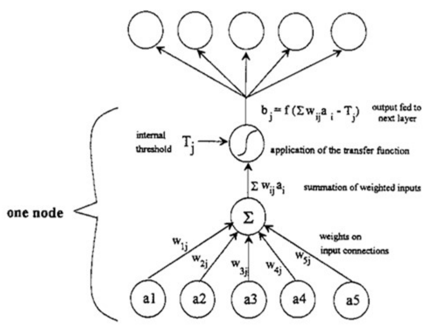


A Gaussian transfer function

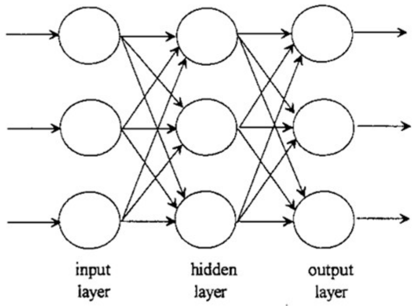
$$f(x) = e^{[-x^2/2]}$$



C. Summary of Node Anatomy



3.2 Topology of a Neural Network

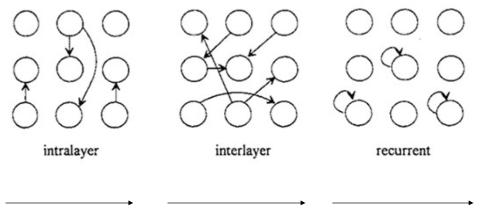


A neural network with one hidden layer

A. Inhibitory or Excitatory Connections

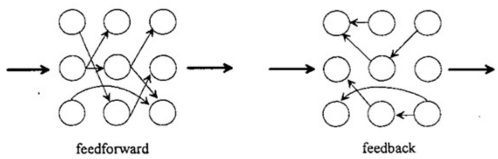
B. Connection Options

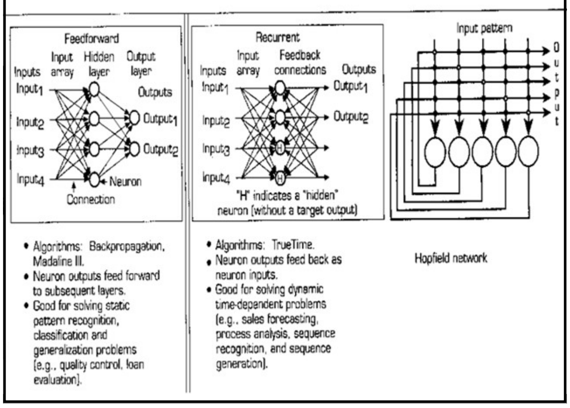
Three connection options in a neural network



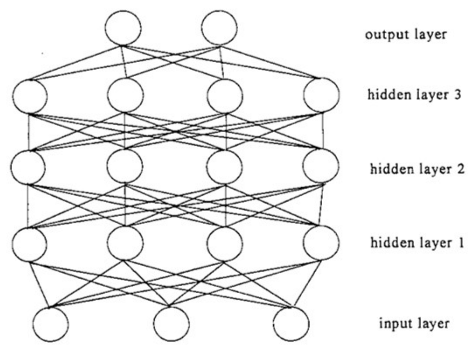
Within the interlayer topology, we have two options:

Feedback connections and feedforward connections





C. Multiple Hidden Layers



3.3 Developing a neural networks

Developing a neural network requires three phases:
- the training or learning phase,
- the recall phase, and
- the generalization phase.

Learning is the actual process of adjusting weight factors based on trial-and-error

A. Stability and Convergence

A globally *stable* neural network maps any set of inputs to a fixed output.

A *convergent* neural network produces accurate input-output relations.

B. Types of Learning

Two major categories based on the input format:

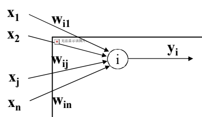
binary-valued input (0s and 1s) or continuous-valued input

Each of these can be further divided into two basic categories:

supervised learning and unsupervised learning

- Supervised learning – An external teacher controls the learning and incorporates global information.
- Unsupervised learning – No external teacher is used and instead the neural network relies upon both internal control and local information.

1. Hebb Learning Rule (unsupervised learning)



$\Delta w_{ij} = w_{ij}(n+1) - w_{ij}(n)$
$= \eta y_i x_j \quad j=1, 2, \ldots, n$
$\eta > 0$ Learning rate
$\Delta w_{ij} = \eta f(w_i^T X)\, x_j$

Example 1

Initial weight vector

$$W^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

Input vector

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Training sample

$$X_1 = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}, \quad X_2 = \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}, \quad X_3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

Assumption 1

$$\eta = 1$$
$$f(net) = \mathrm{sgn}(net) = \begin{cases} 1, & net \ge 0 \\ -1, & net < 0 \end{cases}$$

Step 1

$$net^1 = (W^1)^T X_1 = \begin{bmatrix} 1 & -1 & 0 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = 3$$

f(net)=1

$$W^2 = W^1 + \mathrm{sgn}(net^1) X_1 = W^1 + X_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix}$$

Step 2

$$net^2 = (W^2)^T X_2 = \begin{bmatrix} 2 & -3 & 1.5 & 0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} = -0.25$$

f(net)=-1

$$W^3 = W^2 + \mathrm{sgn}(net^2) X_2 = W^2 - X_2 = \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} = \begin{bmatrix} 1 \\ -2.5 \\ 3.5 \\ 2 \end{bmatrix}$$

Step 3

$$net^3 = (W^3)^T X_3 = \begin{bmatrix} 1 & -2.5 & 3.5 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix} = -3$$

f(net)=-1

$$W^4 = W^3 + \mathrm{sgn}(net^3) X_3 = W^3 - X_3 = \begin{bmatrix} 1 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$$

$$y_1 = \mathrm{sgn}\left((W^4)^T X_1\right) = \mathrm{sgn}\left(\begin{bmatrix} 1 & -3.5 & 4.5 & 0.5 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} \right) = \mathrm{sgn}(14.75) = 1$$

$$y_2 = \mathrm{sgn}\left((W^4)^T X_2\right) = \mathrm{sgn}\left(\begin{bmatrix} 1 & -3.5 & 4.5 & 0.5 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} \right) = \mathrm{sgn}(-7) = -1$$

$$y_3 = \mathrm{sgn}\left((W^4)^T X_3\right) = \mathrm{sgn}\left(\begin{bmatrix} 1 & -3.5 & 4.5 & 0.5 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix} \right) = \mathrm{sgn}(-7.25) = -1$$

Assumption 2 $f(net) = \frac{2}{1+\exp(-net)} - 1$

Step 1 $f(net^1) = 0.905$ $W^2 = W^1 + f(net^1)X_1 = \begin{bmatrix} 1.905 \\ -2.81 \\ 1.357 \\ 0.5 \end{bmatrix}$

Step 2 $net^2 = (W^2)^T X_2 = -0.$ $f(net^1) = -0.932$ $W^3 = W^2 + f(net^2)X_2 = \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$

$f(net^2) = -0.077$

Step 3 $f(net^3) = -0.932$ $W^4 = \begin{bmatrix} 1.828 \\ -3.70 \\ 2.44 \\ -0.783 \end{bmatrix}$

$$f(net^1) = f\left((W^4)^T X_1\right) = f(12.888) = 0.999$$
$$f(net^2) = f\left((W^4)^T X_2\right) = f(-0.0275) = -0.014$$
$$f(net^3) = f\left((W^4)^T X_3\right) = f(-7.3145) = -0.998$$

2. Perceptron Learning Rule (supervised learning)

Learning signal $r = d_i - y_i$ d_i desired output

$\Delta w_{ij} = \eta(d_i - \mathrm{sgn}(w_i^T X))x_j$ $j=1,2,\ldots,n$

Example 1

A group of training input vectors:

$$X_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix}, \quad X_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}, \quad X_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$$

Desired output $d_1 = -1$ $d_2 = -1$ $d_3 = 1$

Assumption $\eta = 0.1$

Initial weight vector $W^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$

Step 1 Select sample X_1 and d_1

$$net^1 = (W^1)^T X_1 = \begin{bmatrix} 1 & -1 & 0 & 0.5 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = 2.5$$

$\because$ $d_1 = -1$

$y_1 = \mathrm{sgn}(2.5) = 1$

$$\therefore \quad W^2 = W_1 + 0.1(d_1 - y_1)X_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + 0.1(-1-1)\begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix}$$

Step 2 Select sample X_2 and d_2

$$net^2 = (W^2)^T X_2 = \begin{bmatrix} 0.8 & -0.6 & 0 & 0.7 \end{bmatrix} \bullet \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix} = -1.6$$

$\because$ $d_2 = y_2 = \mathrm{sgn}(-1.6) = -1$ $\therefore$ $W^3 = W^2$

Step 3 Select sample X_3 and d_3

$$net^3 = (W^3)^T X_3 = \begin{bmatrix} 0.8 & -0.6 & 0 & 0.7 \end{bmatrix} \bullet \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix} = -2.1$$

$\because$ $d_3 = 1$

$y_3 = \mathrm{sgn}(-2.1) = -1$

$$\therefore \quad W^4 = W^3 + 0.1(d_3 - y_3)X_3 = \begin{bmatrix} 0.8 \\ -0.6 \\ 0 \\ 0.7 \end{bmatrix} + 0.1(1+1)\begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix} = \begin{bmatrix} 0.6 \\ -0.4 \\ 0.1 \\ 0.5 \end{bmatrix}$$

Let $W^1 = W^4$, Recycle to step 1

$$net^1 = (W^1)^T X_1 = \begin{bmatrix} 0.6 & -0.4 & 0.1 & 0.5 \end{bmatrix} \bullet \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix} = 0.9$$

From 2.5 reduced to 0.9

3. δ Learning Rule , LMS Rule (supervised learning)

Square Error $E = \frac{1}{2}(d_i - y_i)^2$

d_i *desired output*

y_i *NN output*

$$E = \frac{1}{2}[d_i - f(W_i^T X)]^2$$

$$\frac{\partial E}{\partial w_{ij}} = -(d_i - y_i) f'(W_i^T X) x_j \quad j=1,2,\ldots,n$$

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

$$\Delta W_i = \eta (d_i - y_i) f'(net_i) X$$

$$\Delta w_{ij} = \eta (d_i - y_i) f'(net_i) x_j$$

Example 1

A group of training input vectors:

$$X_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix}, \quad X_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}, \quad X_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$$

Desired output $d_1 = -1$ $d_2 = -1$ $d_3 = 1$

Assumption $\eta = 0.1$

Initial weight vector $W^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$

Transfer function

$$f(net) = \frac{2}{1+\exp(-net)} - 1$$

$$f'(net) = \frac{2\exp(-net)}{[1+\exp(-net)]^2}$$

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

Step 1 Select sample X_1 and d_1

$$net^1 = (W^1)^T X_1 = 2.5$$

$$y_1 = f(net^1) = 0.848$$

$$f'(net^1) = \frac{1}{2}\left[1 - (y_1)^2\right] = 0.140$$

$$W^2 = \eta(d_1 - y_1) f'(net^1) X_1 + W^1 = \begin{bmatrix} 0.974 & -0.948 & 0 & 0.526 \end{bmatrix}^T$$

Step 2 Select sample X_2 and d_2

$$net^2 = (W^2)^T X_2 = -1.948$$

$$y_2 = f(net^2) = -0.75$$

$$f'(net^2) = \frac{1}{2}\left[1 - (y_2)^2\right] = 0.218$$

$$W^3 = \eta(d_2 - y_2) f'(net^2) X_2 + W^2 = \begin{bmatrix} 0.974 & -0.956 & 0.002 & 0.531 \end{bmatrix}^T$$

Step 3 Select sample X_3 and d_3

$$net^3 = (W^3)^T X_3 = -2.46$$

$$y_3 = f(net^3) = -0.842$$

$$f'(net^3) = \frac{1}{2}\left[1 - (y_3)^2\right] = 0.145$$

$$W^4 = \eta(d_3 - y_3) f'(net^3) X_3 + W^3 = \begin{bmatrix} 0.947 & -0.929 & 0.016 & 0.505 \end{bmatrix}^T$$

4. Widrow-Hoff Learning Rule (supervised learning)

Special δ Learning Rule ($f(W_i^T X) = W_i^T X$)

$$net_i = W_i^T X$$

$$r = d_i - W_i^T X$$

$$\Delta w_{ij} = \eta (d_i - W_i^T X) x_j \quad j=1,2,\ldots,n$$

5. Correlation Learning Rule (supervised learning)

Special Hebb Learning Rule (binary function and $y_i = d_i$)

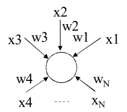
$$\Delta w_{ij} = c d_i x_j \qquad j = 1, 2, \cdots, n$$

6. Winner-Take-All Learning Rule (unsupervised learning)

$$W_m^T X = \max_{i=1,2,\ldots,p} (W_i^T X)$$

$\Delta w_{mj} = \alpha(x_j - W_{mj})$ j=1,2,…,n ; α>0 learning constant

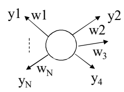
7. Star Learning Rule (supervised learning)



Inner-Star Training Algorithm

$w_i(n+1) = w_i(n) + \eta[x_i - w_i(n)]$

η Training rate coefficient



Outer-Star Training Algorithm

$w_i(n+1) = w_i(n) + \beta[y_i - w_i(n)]$

β Training rate coefficient

8. Gradient Descent Algorithm (supervised learning)

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

E is the error function

η is the learning rate

9. Stochastic Training Algorithm

We accept a random weight change if it reduces the output error vector,ε. If the change increases ε, we generally reject the change.

10. Simulated Annealing Algorithm

Gauss Function : $G_g(x) \approx \exp[-x^2/T(t)]$

$T(t+1)=T(0) / \ln(t+1)$

C. Checking the Performance of the Neural Network

Two main steps:

1. Recall step

How well does the neural network recall the predicted responses (output vector) from data sets used to train the network.

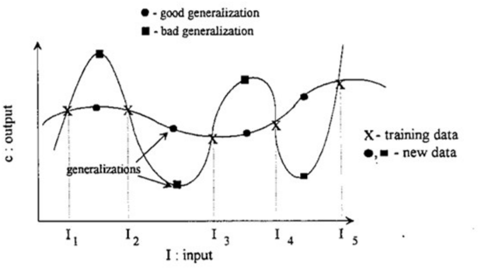
A well-trained network should be able to produce an output with very little error from the desired output.

2. Generalization step

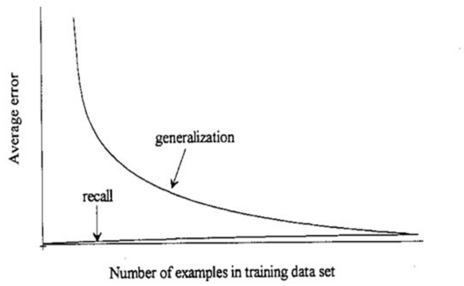
How well does the network predict responses from data sets that were not used in training.

A well-trained network should provides input-output mapping with good generalization capability.

An illustration of good and bad generalizations by a trained neural network



An typical learning curve for a well-trained neural network



Chapter Highlights

1. **The basic network component is the node (processing element) that has an n-dimensional input vector, an internal threshold value, and n weight factors, which multiply all inputs.**
2. **The node transfers the input to the output through the weight factors and a transfer function.**
3. **The topology of a neural network refers to how its nodes are interconnected. (by organizing the nodes into layers, connecting them, and weighting the interconnections.)**
4. **Developing a neural network requires three phases: the training or learning phase, the recall phase, and the generalization phase.**
5. **Learning is the actual process of adjusting weight factors based on trial-and-error. "given enough parameters, you can fit an elephant's back."**

Questions for Review

1. **What determines the output from a node?**
2. **What functional form do we choose for transfer function?**
3. **List the three options for connecting nodes to one another.**
4. **Describe the two basic approaches to training neural networks.**
5. **Analyze the inhibitory and excitatory connections.**

Questions for Discussion

1. **Why are learning algorithms important to an ANN?**
2. **Explain how ANN's learn in a supervised and in an unsupervised mode.**
3. **Explain the difference between a training set and a testing set. Can the same set be used for both purposes? Why or why not?**
4. **Discuss the relationship between stability and convergence of the neural network.**
5. **Compare the learning algorithms and explain how learning (training) is executed.**