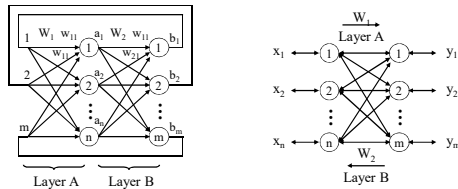


5.2 BAM Network

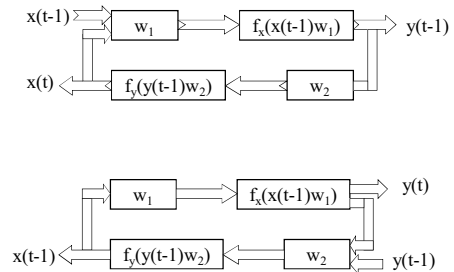
Bidirectional Associative Memory



$$x(t) = f_y(f_x(x(t-1)W_1)W_2)$$

$$y(t) = f_x(f_y(y(t-1)W_2)W_1)$$

Calculating process



$$(x_i, y_i), i=1,2,\dots,P$$

Learning rule

Assuming: $x \in \{-1, +1\}^N$, $y \in \{-1, +1\}^M$

Hebb learning formula $W_1 = \sum_{k=1}^P x^k y^k$ $W_2 = \sum_{k=1}^P y^k x^k$ $W_1 = W_2^T$

Definition of energy function:

$$E(x, y) = -\frac{1}{2} x^T W_1 y^T - \frac{1}{2} y^T W_2 x^T$$

$$= -x^T W_1 y^T$$

$$\Delta E = -\Delta x^T W_1 y^T$$

$$= -\sum \Delta x_i \sum y_j w_{ij}$$

$$= -\sum \Delta x_i y_i w_{ii}$$

$$\text{If } \Delta x_i > 0 \text{ then } y_i w_{ii} > 0$$

$$\text{If } \Delta x_i < 0 \text{ then } y_i w_{ii} < 0$$

$$\text{Therefore, } \Delta E < 0$$

w_i is the i^{th} row weight factors in the W
 w^j is the j^{th} column weight factors in the W

Example 1

$$x_1(1 \ 0 \ 1 \ 0 \ 1), \quad y_1(1 \ 1 \ 1 \ 1)$$

$$x_2(1 \ 0 \ 1 \ 0 \ 0), \quad y_2(0 \ 1 \ 1 \ 0)$$

$$x_3(0 \ 1 \ 0 \ 1 \ 1), \quad y_3(1 \ 0 \ 0 \ 1)$$

Change the vectors to $\{-1, +1\}$

$$x_1'(1 \ -1 \ 1 \ -1 \ 1), \quad y_1'(1 \ 1 \ 1 \ 1)$$

$$x_2'(1 \ -1 \ 1 \ -1 \ -1), \quad y_2'(-1 \ 1 \ 1 \ -1)$$

$$x_3'(-1 \ 1 \ -1 \ 1 \ 1), \quad y_3'(1 \ -1 \ -1 \ 1)$$

$$W = \sum_{k=1}^3 (x_k')^T y_k' = \begin{bmatrix} -1 & 3 & 3 & -1 \\ 1 & -3 & -3 & 1 \\ -1 & 3 & 3 & -1 \\ 1 & -3 & -3 & 1 \\ 3 & -1 & -1 & 3 \end{bmatrix}$$

$$\text{sgn}(x, W) = \text{sgn}(1 \ 5 \ 5 \ 1) = y_1$$

$$\text{sgn}(y, W^T) = \text{sgn}(4 \ -4 \ 4 \ -4)$$

$$= (1 \ -1 \ 1 \ -1)$$

change to binary

$$\text{sgn}(y_1 W^T) = (1 \ 0 \ 1 \ 0) = x_1$$

$$\delta = (0 \ 1 \ 0 \ 0 \ 0)$$

$$x_1 + \delta = (1 \ 1 \ 1 \ 0 \ 1)$$

$$\text{sgn}[(x_1 + \delta)W] = \text{sgn}[2 \ 2 \ 2 \ 2]$$

$$= y_1$$

Chapter 6 ANN Hybrid System

6.1 Fuzzy Neural Network

6.2 Evolutionary Computation and Genetic Algorithm

6.3 A Hybrid System of Expert Systems and Neural Networks

6.1 Fuzzy Neural Network

Neural-fuzzy networks implements fuzz-logic inferencing through neural networks.

1. Fuzzy-logic Systems

Fuzzy logic grew out of a desire to quantify rule-based systems. It provides a way to quantify certain quantifiers such as *approximately, often, rarely, several, few, and very*.

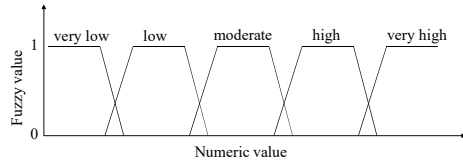
Relationship of fuzzy-logic systems to the two main areas of AI

knowledge type	Information framework	
	symbolic	numerical
structured (based on rules)	expert system	fuzzy-logic system
unstructured	—	neural network

A. Fuzzy set

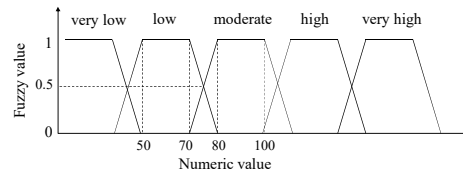
quantify the *degree of membership* with values between 0 (not a member) and 1 (definitely a member).

The following figure shows a representation of energy requirement in fuzzy terms.



B. Conversion Between Numeric and Fuzzy-Logic Variables

How to convert a numeric variable to a fuzzy-logic variable through *fuzzifier*, and to convert the fuzzy-logic variable back to a numeric variable through a *defuzzifier*.

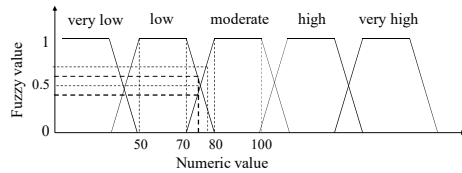


The numeric variable are denoted $numeric(x)$ and the fuzzy-logic variables are denoted $fuzzy(e.g., 50 \text{ to } 70 \text{ for low, and } 80 \text{ to } 100 \text{ for moderate})$.

$$numeric(60) = fuzzy(0, 1, 0, 0, 0)$$

$$numeric(90) = fuzzy(0, 0, 1, 0, 0)$$

For numeric values in the transition regions, we use a linear interpolation between the beginning and ending values of the region.



$$numeric(74) = fuzzy(0, \frac{80-74}{80-70}, \frac{74-70}{80-70}, 0, 0) = fuzzy(0, 0.6, 0.4, 0, 0)$$

Similarly, we can convert the fuzzy-logic variable back to a numeric variable using the exact opposite process.

$$fuzzy(0, 0.3, 0.7, 0, 0) = numeric(0.3 = \frac{80 - x_1}{80 - 70}) \quad \text{or} \quad numeric(0.7 = \frac{x_2 - 70}{80 - 70})$$

$$x_1 = 77 \quad x_2 = 77$$

C. Union and Intersection of Fuzzy sets

Define two fuzzy sets:

$$I = \{i_1/x_1, i_2/x_2, \dots, i_n/x_n\}$$

$$J = \{j_1/x_1, j_2/x_2, \dots, j_p/x_p\}$$

where x_1, x_2, \dots are members of the set with degrees of membership i_1, i_2, \dots (for set I) and j_1, j_2, \dots (for set J).

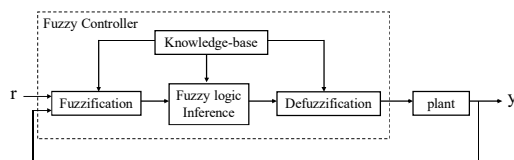
Union of two fuzzy sets:

$$I \cup J = \{(\max(i_1, j_1))/x_1, (\max(i_2, j_2))/x_2, \dots\}$$

Intersection of two fuzzy sets:

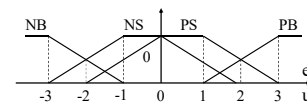
$$I \cap J = \{(\min(i_1, j_1))/x_1, (\min(i_2, j_2))/x_2, \dots\}$$

D. Fuzzy Control



Fuzzy-logic variable (e,u)

	-3	-2	-1	0	1	2	3
PB	0	0	0	0	0	0.5	1
PS	0	0	0	0	1	0.5	0
0	0	0	0.5	1	0.5	0	0
NS	0	0.5	1	0	0	0	0
NB	1	0.5	0	0	0	0	0



$$NS = \frac{0.5}{-2} + \frac{1}{-1}$$

Control rule table

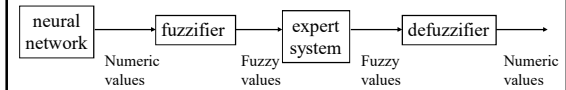
e	NB	NS	0	PS	PB
u	PB	PS	0	NS	NB

$$R = (NB_e * PB_u) + (NS_e * PS_u) + (0_e * 0_u) + (PS_e * NS_u) + (PB_e * NB_u)$$

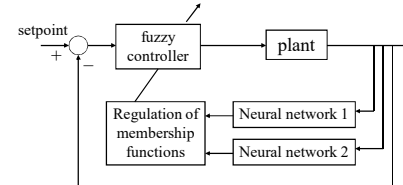
$$u = e \circ R$$

2. Neural-Fuzzy Networks

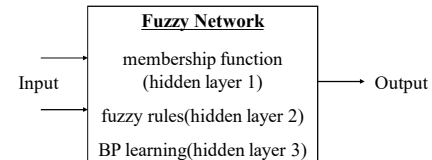
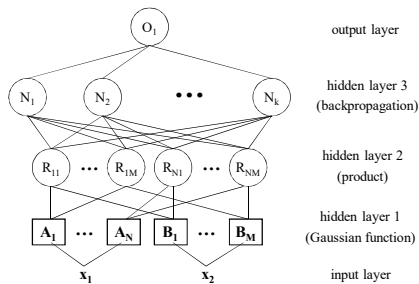
A. Neural-Fuzzy Networks for Expert Systems



B. The architecture of the neural-fuzzy controller



3. Fuzzy Neural Network



Hidden layer 1:

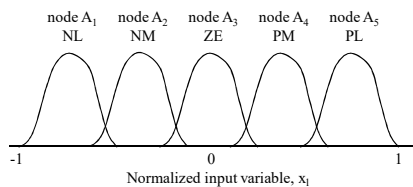
represents a radially symmetric (Gaussian) membership functions of A_i ($i=1$ to N) and B_j ($j=1$ to M) for the normalized input variables x_1 and x_2 , respectively.

\bar{x} is the center and σ is the standard deviation of a Gaussian membership function.

$$O_{A_i}^1 = \exp \left[- \left(\frac{x_1 - \bar{x}_1}{\sigma_{a1}} \right)^2 \right]$$

$$O_{B_j}^1 = \exp \left[- \left(\frac{x_2 - \bar{x}_2}{\sigma_{b1}} \right)^2 \right]$$

A common representation of the Gaussian membership function stored in hidden layer 1 of a fuzzy network.



Hidden layer 2:

Taking the dot product of the outputs of the membership functions(hidden layer 1) $O_{A_i}^1$ and $O_{B_j}^1$:

$$O_l^2 = O_{A_i}^1 \bullet O_{B_j}^1 = \min(O_{A_i}^1, O_{B_j}^1) \quad (l = 1, M * N)$$

For example, the output of the first node(R_{11}) in the hidden layer 2 reflects whether the rule of $A_1(x_1$ is negative large) and $B_1(x_2$ is negative large) should be applied.

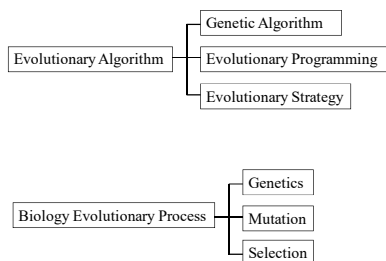
Hidden layer 3:

operates the same as a standard backpropagation network, having weight factors (w_{ij}) and a sigmoid transfer function. The weight factors are adjusted to map the firing strength of the fuzzy rules onto the desired output variable.

Hidden layer 4:

operate the same as a standard backpropagation network, as in layer 3.

6.2 Evolutionary Computation and Genetic Algorithm



GA basic operation

Step 1: Initiation

Step 2: Selection (reproduction)
e.g. using fitness function

Step 3: Crossover

Example 1

A	1001111	⇒	1001000	C
B	0011000		0011111	D

Example 2

A:	001111
B:	111100
Shield	010101
A':	011110
B':	101101

Step 4: Mutation 0→1
 1→0

Practice

Max { $f(x)=x^2$ }, $x=0 \sim 31$

1. Initiation (binary code)

标号	串	X	适配值	占整体百分数
1	01101	13	169	14.4
2	11000	24	576	49.2
3	01000	8	64	5.5
4	10011	19	361	30.9
总计 (初始种群整体)			1170	100

2. Selection(reproduction)

第一代群体的选择 (复制)

标号	初始群体	X	$f(x)=x^2$	选择复制的概率	期望复制数	实际得到复制数
1	01101	13	169	0.14	0.58	1
2	11000	24	576	0.49	1.97	2
3	01000	8	64	0.06	0.22	0
4	10011	19	361	0.31	1.23	1
总计			1170	1.0	4.0	4
平均			293	0.25	1.0	1
最大值			576	0.49	1.97	2

$$169/1170=0.14 \quad 169/293=0.58$$

3. Crossover

复制交叉后的各项数据

新标号	复制后的匹配池	匹配对象 (随机选取)	交叉点 (随机取)	新种群	X值	$f(x)=x^2$
1	01101	2	4	01100	12	144
2	11000	1	4	11001	25	625
3	11000	4	2	11011	27	729
4	10011	3	2	10000	16	256
总 计						1754
平 均						439
最大值						729

平均值 293 → 439
最大值 576 → 729

4. Mutation

取变异概率0.001, 种群共有4*5=20位, 20*0.001=0.02位, 所以本例无串位值的改变。至此, 完成一代遗传。

Application in ANN

用遗传算法学习神经网络权重 w_{ij}

1. 二进制编码

W字符串表示的值和实际数值间的关系:

$$w(i, j) = w_{\min}(i, j) + \frac{\text{binary}(t)}{2^n - 1} [w_{\max}(i, j) - w_{\min}(i, j)]$$

如w在1~0间, 4位二进制: $w = 0 + \frac{1111}{2^4 - 1} [1 - 0] = 1$

W学习过程: 交叉、变异, 产生下一代网络。

2. Real number code

* code (0.4, -0.3, 2.1, 1.3, 0.9, -0.6, 4.5, -0.1, 0.7)

* estimate function $f=1/\sum e_i^2$

* Initiation

determine the weight factors at random

(0.4, -0.3, 2.1, 1.3, 0.9, -0.6, 4.5, -0.1, 0.7)

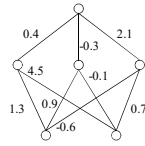
(0.7, -0.9, 1.2, 0.8, 1.4, 0.1, -1.1, 0.2, -1.1)

* crossover (0.4, -0.9, 1.2, 1.3, 1.4, 0.1, 4.5, 0.2, -1.1)

* mutation (0.4, -0.3, 2.1, 1.3, 0.9, -0.6, 4.5, -0.1, 0.7)

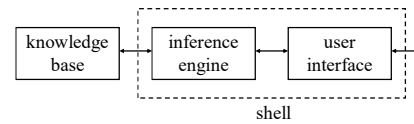
stochastic value -1.1 stochastic value -0.7

(0.4, -0.3, 1.0, 1.3, 0.9, -0.6, 4.5, -0.8, 0.7)



6.3 A Hybrid System of Expert Systems and Neural Networks

1. The structure of an expert system



knowledge in expert systems

Knowledge = facts + rules + heuristics

- shell
1. Explains how the system reaches a conclusion
 2. Explains why it needs certain information
 3. Adds information to the knowledge base

2. Neural networks versus expert systems

Neural networks	Expert systems
example-based	rule-based
domain-free	domain-specific
finds rules	needs rules
little programming	much programming
easy to maintain	difficult to maintain
fault-tolerant	not fault-tolerant
needs a database	needs a human expert
fuzzy logic	rigid logic
adaptive system	requires reprogramming

3. Advantages and limitations of expert systems and neural networks

	Expert systems	Neural networks
very effective	In applying a fixed set of facts, rules and heuristic(i.e., knowledge) to a domain-specific problem, typically involving only simple mathematics.	In organizing and detecting patterns from unpredictable and/or imprecise input data, in learning by examples, and in generalizing to new situations.
less effective	In processing raw sensory data from the real, unpredictable world.	In providing in-depth solutions and full understanding of each problem as well as the reasoning behind its solution(i.e., no explanation capability).

4. An illustration of an expert network

