Soft computing

* genes → chromosome → genotype → phenotype

* Genotype → values ترميز | TRUE [collection of genes] ← chromosome

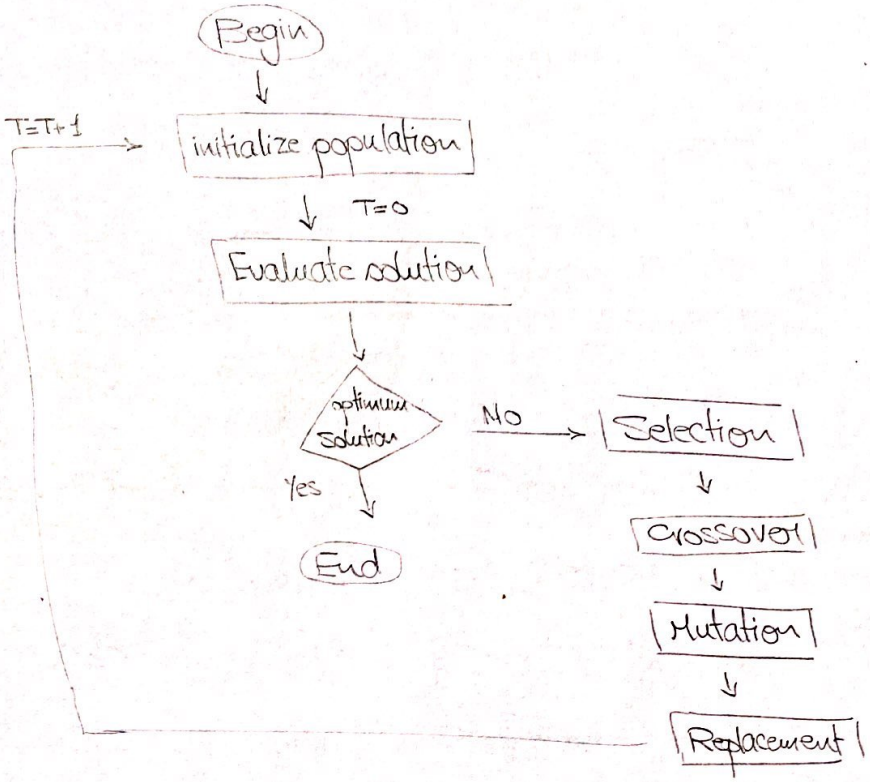
Phenotype → action | True [collection of aspects]

genotype contains all information to construct the phenotype

(1) Blind generate & test
(2) GA
↳ natural selection (select the best, discard the rest)

* Genetic algorithm are search & optimization techniques

→ Flowchart



→ Pseudo code

canonical

```
SimpleGeneticAlgorithm()
{
  Initialize population;
  calculate fitness;
  while (fitness value != optimal value)
  {
    Selection;
    crossover;
    mutation;
    replacement;
    calculate fitness
  }
}
```

```
canonical_GA()
{
  = G0;
  for(i=1 to Max-Gen)
  { Evaluate ... }
}
```

Enviroment → problem
population → solutions [search space / state space]
individual → one solution
fitness → Quality of a solution
chromosome → format of solution
gene → part of solution
reproduction → crossover
copying error → mutation

When to stop:
(1) Best fitness
(2) Number of generations [Max]
(3) No new chromosome
(4) No improvements after a number of generations

Phenotype ⇄ genotype (encoding / decoding)

Scanned with CamScanner

64	32	16	8	4	2	1

* Crossover: [Exploration]
↳ good bits from different parents

~> P_c: probability that crossover will occur ↳ given

~> $r_1 \in [1, L-1]$
↳ crossover point [after how many genes]

~> $r_2 \in [0, 1]$
↳ if $r_2 \leq P_c$ then perform crossover

- **Single point crossover** ~> used in GP [positional biased]

|1|0|1|0|0| → |1|0|0|1|1|

|0|1|0|1|1| → |0|1|1|0|0|

- **N point crossover** (some positional biased)

|1|0|1|0|0| → |1|0|0|1|0|

|0|1|0|1|1| → |0|1|1|0|1|

- **order-1 crossover** [permutation]
↳ to preserve relative order

- **Uniform crossover**

يعتبروا نفس الحاجة

* Mutation: [Exploitation] [flipping bits]

~> P_m: probability that mutation will occur

~> $r \in [0, 1]$
↳ if $r \leq P_m$ then perform mutation.

Salesman problem
- permutation encoding
- Tournament selection
- order 1 crossover
- Swap mutation

* Gray code representation: [change in 1 bit]
↳ small changes in the genotype causes small changes in phenotype ↳ (unlike binary)
↳ it gives stability in exploring the search space.

* fitness
↳ Max
↳ average

* Selection:

1. Roulette wheel Selection:
↳ better individuals get higher chance & chances proportional to fitness.

↳ Maximization: $\frac{\text{fitness}}{\text{total fitness}}$

↳ Minimization: $\frac{\frac{1}{F}}{\text{total fitness}}$ or $\frac{100-F}{\text{total fitness}}$

↳ Roulette is fitness biased.

↳ suffers from premature convergence

2. Tournament selection
↳ 2 solutions and the better solution is placed in mating pool

3. Rank selection
↳ relative rather than absolute fitness
↳ worst ~> 1^{st}
second ~> 2^{nd}
↳ الفرق صغير بين الgenes
↳ cumulative fitness
↳ roulette wheel

Scanned with CamScanner

* Encoding — على حسب الـ encoding ، الـ crossover والـ mutation بيتأثروا بس الـ Selection لأ. independent

↝ permutation encoding ↝ بنحافظ على الترتيب وعدم التكرار

↳ single point may produce some invalid offsprings

| 1 | 2 | 7 | 8 | ↝ genotype 1, 2, 7, 8

↳ integer encoding ↝ genotype 0 → ∞

① population using integer chromosome

↳ | 1 | 9 | 8 | 100 | or | 1 | 5 | 8 | 12 |

② population using permutation مختلفين نفس الأرقام بس ترتيب

↳ | 7 | 8 | 2 | 1 | or | 7 | 2 | 1 | 8 |

* Crossover for permutation lead to inadmissible solutions

* Mutation for permutation — لـ حالة الـ order-1

1. Insert ↳ Bit-flipping mutation { creep ↝ تجيب رقم وتقلله أو تزوده براحتك ; random ↝ تجيب رقم عشوائي }

↳ move the second to follow the first, shifting the rest along

2. Inversion

↳ pick two alleles (genes) at random and then invert the substring between them

3. Swap

↳ pick two alleles (genes) at random and swap their position

* Mutation over floating point chromosome:

1. Uniform

2. Non-Uniform

* Replacement: [cutoff selection & mixing strategy] ↳ weak half ↝ compare with # ↝ strong & weak

1. Generational replacement (GGA) ↝ used in GP

↳ entire set of parents is replaced by the offspring

2. Steady-state replacement (SSGA) [next generation]

↳ specific number(k) of individuals are selected for reproduction

3. Elitist Strategy (Elitism) [next generation]

↳ Best-so-far individuals (same as steady-state)

* population models

1. Gap — الى اتغير

↝ GGA = 1

↝ SSGA = k / Popsize

2. Overlap — الى ثابت

↝ SSGA = (popsize − k) / Popsize

↝ 1 − Gap

⟹ Mutation only is possible: ↝ different combinations

↳ as the mutation can bring genes that won't happen using crossover

↳ introduce new genes

Scanned with CamScanner

* Mutation over FP chromosomes

1. Uniform FP mutation:

↳ $r_1 \in [0, 1]$ [This means equal chance to go left or right]

$\Delta = \Delta_L$ if $r_1 \leq 0.5$ ⇝ $\Delta_L = x_i - LB_i$

$\Delta = \Delta_U$ if $r_1 > 0.5$ ⇝ $\Delta_U = UB_i - x_i$

↳ $r_2 \in [0, \Delta]$

$\Delta = \Delta_L$ then $x_{new} = x_i - r_2$

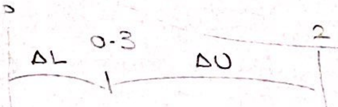
$\Delta = \Delta_U$ then $x_{new} = x_i + r_2$

Ex:

0.3	0.5	0.7	0.9

$r1 = 0.7$

$r2 = 0.3$



$r1 > 0.5$ ⇝ ΔU

$x_{new} = 0.3 + 0.3$

$= 0.6$

$\Delta L = 0.3 - 0 = 0.3$

$\Delta U = 2 - 0.3 = 1.7$] [illegible]

$r1 = 0.2$

$r2 = 0.25$

$r1 \leq 0.5$ ⇝ ΔL

$x_{new} = 0.3 - 0.25$

$= 0.05$

---

2. Non uniform mutation:

↳ $r_1 \in [0, 1]$

$y = \Delta L$ if $r_1 \leq 0.5$

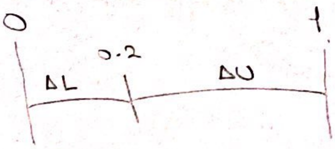
$y = \Delta U$ if $r_1 > 0.5$

↳ $\Delta(t, y) = y\left(1 - r^{\left(1 - \frac{t}{T}\right)^b}\right)$

Ex:

0.5	0.3	0.7	0.2

↑ b ⇝ ↓ mutation



$\Delta L = 0.2 - 0 = 0.2$

$\Delta U = 1 - 0.2 = 0.8$

$r1 = 0.6$ $\quad r = 0.1$ (random) (given)

$0.6 > 0.5$ ⇝ $\Delta U = 0.8 = y$

---

t = 0

$\Delta = y\left[1 - r^{\left(1 - \frac{t}{T}\right)^b}\right]$ (b ⇝ given; T ⇝ given)

$= 0.8\left[1 - 0.1^{\left(1 - \frac{0}{10}\right)^1}\right]$

$= 0.72$

$x_{new} = 0.2 + 0.72$

$= 0.92$

↳ max mutation

t = 10

$\Delta = 0.8\left[1 - 0.1^{\left(1 - \frac{10}{10}\right)^1}\right]$

$= 0$

$x_{new} = 0.2 + 0 = 0.2$

no mutation ↲

t = 5

$\Delta = 0.8\left[1 - 0.1^{\left(1 - \frac{5}{10}\right)^1}\right]$

$= 0.54$

$x_{new} = 0.2 + 0.54$

$= 0.74$

Scanned with CamScanner

* **Island genetic algorithm** ⤳ Static: good migrant / randomly selected migrant — bad migrant / randomly selected
  ↳ Dynamic: migration → probability
  destination → random (acceptance strategy)
  ↳ fitness > average fitness of island
- ↳ run same problem
- ↳ Migration ← Chromo تبادل ال
- ↳ can be easily parallelized
- ↳ communication / migration rate / selection / replacement

* **Schema theory:**
  - ↳ order $o(S)$ → fixed positions
  - ↳ length $d(S)$ ⤳ distance between first + last fixed positions

$011*1011**$ — $2^3 = 8$ solutions

1. Selection

- Average fitness of population: $\frac{\sum_{i=1}^{Popsize} f_i}{Popsize} = \bar{F}$ ⟶ $F = \bar{F} \times popsize$ (where $\sum_{i=1}^{Popsize} f_i = F$)
- Average fitness of matching schema: $\frac{\sum_{i=1}^{P} f_j}{P} = f_{av}(S)$
- F = total fitness of population, $F = \sum_{i=1}^{Popsize} f_i$
- $m(S,t+1) = m(S,t) * popsize * \frac{f_{av}(S)}{F}$

$m(S,t+1) = m(S,t) * popsize \times \frac{f_{av}(S)}{\bar{F} \times popsize}$

* $m(S,t+1) = m(S,t) * \frac{f_{av}(S)}{\bar{F}}$

$\frac{f_{av}(S)}{\bar{F}} > 1$

$f_{av}(S) > \bar{F}$ ⤳ $f_{av}(S) = \bar{F} + \epsilon \bar{F} = (1+\epsilon)\bar{F}$

$m(S,t+1) = m(S,t) * \frac{(1+\epsilon)\bar{F}}{\bar{F}}$

$= m(S,0) * (1+\epsilon)^t$

Generation (iterations)
- population
  - ↳ chromosome
    - ↳ genes

GP:
- * functions & terminals
- * GA → linear / fixed size
- GP → non-linear / vary
- * The key lies in that GP uses a fitness measure to determine which functions survive to reproduce in each generation

Next gen. ⟵ Yes — $i = pop\ size$?
↓
Select variation
- Pm → select 1 → mutation → add off → $i = i+1$
- Pc → select 2 → selection → add off → $i = i+2$
(→ back to $i = pop\ size$?)

2. Crossover

$P_s = 1 - P_c \frac{d(S)}{l-1}$ $\quad$ $P_d = \frac{d(S)}{l-1}$

3. Mutation

$P_s = (1-P_m)^{o(S)}$ or $1 - o(S) P_m$

$P_d = P_m$

$P_s = 1 - P_d = 1 - P_m$

* Reproductive Schema Growth Equation (RSG)

$$m(S,t+1) = m(S,t) * \frac{f_{av}(S)}{\bar{F}} * \left(1 - pc \frac{d(S)}{l-1} - o(S) P_m\right)$$

Scanned with CamScanner

Fuzzy Logic:

* Boolean logic ⇝ sharp distinctions. [two-valued]
* Fuzzy logic ⇝ logic that extends the range of truth values to all real numbers in the interval between 0 and 1. [multi-valued]
  ⇝ reflects how people think [can handle the concept of partial truth]
  ⇝ knowledge representation
  ⇝ accepts noisy, imprecise input

Fuzzy sets: ← fuzzy variables

↳ can represent the degree to which quality is possessed
↳ x-axis → universe of discourse [range of all possible value]
↳ y-axis → membership value

*characteristic function
↳ $f_A(x): x \to \{0,1\}$

* membership function
↳ $m_A(x): x \to [0,1]$

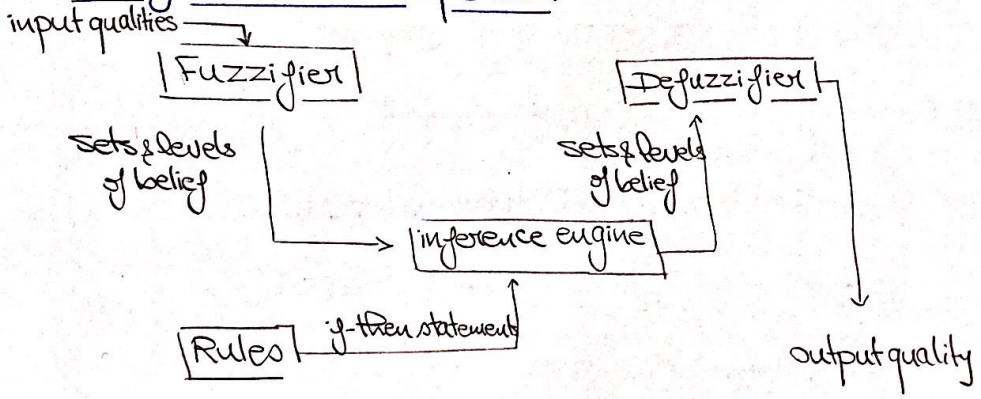
* linguistic variable = fuzzy variable
• triangle $\{a,b,c\} = \{0,1,0\}$
• trapezoidal $\{a,b,c,d\} = \{0,1,1,0\}$

* Complement: $1 - \mu_A(x)$
* Intersection: min / AND = conjunction
* Union: max / OR = disjunction

? rules من أين نأخذ ال
- customer
- expert
- GA

*antecedents (x) → المتغير
* consequent (y) → المتغير

* Fuzzy control component.



① Fuzzification
② Evaluation of Rules
③ De-fuzzification

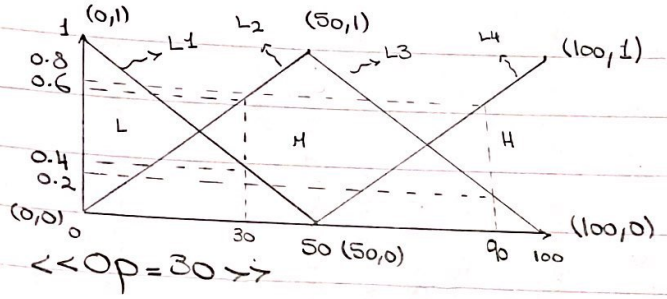
---

crisp data
↓
Fuzzifier
↓
Fuzzy rules
↓
Fuzzy output set
↓
Defuzzifier
↓
crisp data

* combining neural network with fuzzy logic reduces time to establish rules by analyzing clusters of data

Scanned with CamScanner

OP/Pdp



<<Op = 30>>

* L_1 : (0,1) (50,0)

$a = \text{slope} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0-1}{50-0} = \frac{-1}{50}$

* Substituting with (0,1) $\rightsquigarrow$ $1 = \frac{-1}{50} \times 0 + b$

$1 = 0 + b \rightsquigarrow b = 1$

* Equation of L_1 $\rightsquigarrow$ $y = \frac{-x}{50} + 1$
* Substituting with op = 30 $\rightsquigarrow$ $y = \frac{-30}{50} + 1 \rightsquigarrow y = \underline{\underline{0.4}}$ [membership value]

* L_2 : (0,0) (50,1)

$a = \text{slope} = \frac{1-0}{50-0} = \frac{1}{50}$

* substituting with (0,0) $\rightsquigarrow$ $0 = 0 + b \rightsquigarrow b = 0$
* Equation of L_2 $\rightsquigarrow$ $y = \frac{x}{50}$
* Substituting with op = 30 $\rightsquigarrow$ $y = \frac{30}{50} \rightsquigarrow y = \underline{\underline{0.6}}$ [membership value]

<<pdp = 90>>

* slope of $L_3 = \frac{0-1}{100-50} = \frac{-1}{50}$ $\rightsquigarrow b = 2$
* substituting with pdp = 90 $\rightsquigarrow$ $y = \frac{-90}{50} + 2 = 0.2$

* slope of $L_4 = \frac{1-0}{100-50} = \frac{1}{50}$ $\rightsquigarrow b = -1$
* substituting with pdp = 90 $\rightsquigarrow$ $y = \frac{90}{50} - 1 = 0.8$

$\Rightarrow$ RULES:

R1: 0.4 AND 0 $\rightsquigarrow$ 0

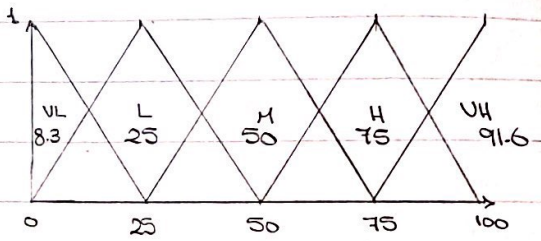
R2: 0.4 OR 0 $\rightsquigarrow$ 0.4 (L)

R3: 0.6 AND (1-0.8) $\rightsquigarrow$ 0.2 (M) (0.2)

R4: 0 OR 0.8 $\rightsquigarrow$ 0.8 (H)

R5: 0 AND 0.8 $\rightsquigarrow$ 0

CP



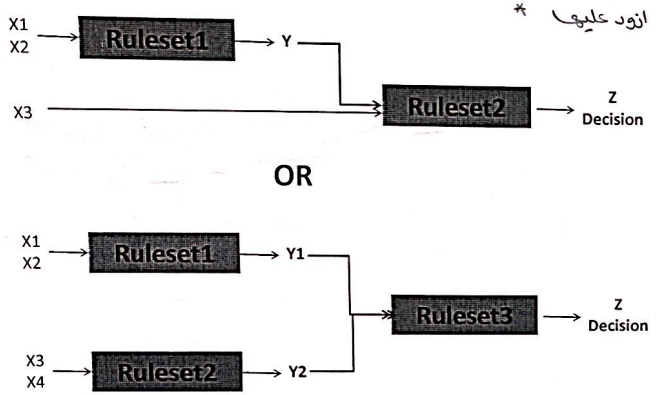
$\rightsquigarrow$ Centroid of L = 0+25+50/3 = 25

$$\frac{25 \times 0.4 + 50 \times 0.2 + 75 \times 0.8}{0.4 + 0.2 + 0.8} = 57.1$$

Scanned with CamScanner

Multiple Rulesets

* Max
* average
* ازود عليهم



12/7/2021 37

Example

- ***Given a medical information system with governing variables X1, X2, X3. It is required to predict decision D which is Benign or Malignant B*** {0, 0, 10}, **M** {0, 10, 10}.

The following information is given: X1, X2, and X3 have **3** fuzzy sets **L, M, H** (range 0 - 100)

L {0, 0, 50}, medium **M** {0, 50, 100}, good **H** {50, 100, 100}

With the following rulesets:

RS1:

R1: If X1=L **AND** X2=M THEN Y=L 0.4

R2: If X1=M **AND** X2 =H THEN Y=H 0.6

RS2:

R3: If Y=L **AND** X3=M THEN D=B 0.2

R4: If Y=H **AND** X3=H THEN D=M 0.6

Using triangular membership function, Predict D for X1 = 30 , X2 = 80 and X3 = 90 ?

12/7/2021 38

ي $x_1 = 30$:

- Slope $= \frac{1-0}{0-50} = \frac{1}{-50}$

L_1: $y = \frac{1}{-50}x + b$

$1 = 0 + b \rightsquigarrow b = 1$

$y = \frac{30}{-50} + 1 = 0.4$

- Slope $= \frac{1-0}{50-0} = \frac{1}{50}$

L_2: $y = \frac{x}{50}$

$y = \frac{30}{50} = 0.6$

ي $x_2 = 80$

- Slope $= \frac{0-1}{100-50} = \frac{-1}{50}$

L_3: $y = \frac{-x}{50} + 2$

$y = \frac{-80}{50} + 2 = 0.4$

- Slope $= \frac{1-0}{100-50} = \frac{1}{50}$

L_4: $y = \frac{x}{50} - 1$

$y = \frac{80}{50} - 1 = 0.6$

ي $x_3 = 90$

L_3: $y = \frac{-90}{50} + 2$

$= 0.2$

L_4: $y = \frac{90}{50} - 1$

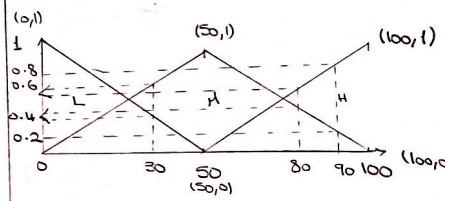
$= 0.8$

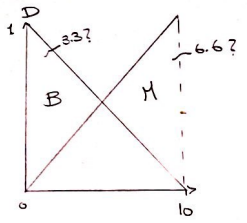
$D = M$ [0.4] ?

$D = B$ [0.2]

$D = B$ [0.6]

$D = \frac{0.4\overline{M} + 0.2\overline{B} + 0.6\overline{B}}{0.4 + 0.2 + 0.6}$





Scanned with CamScanner

* Defuzzification techniques:

① Max membership principle: [height method]

↳ [illegible] point [illegible] 1 [illegible] Z [illegible]

↳ if multiple peaks exist, the middle one is chosen.

② Mean max membership: [middle of maxima]

↳ $Z = \frac{a+b}{2}$ (average)

③ centroid method: [center of area | center of gravity]

↳ area under graph

④ weighted average method

↳ $\frac{\text{centroid}_1 \times \text{membership}_1 + \text{centroid}_2 \times \text{membership}_2 + \dots}{\text{membership}_1 + \text{membership}_2 + \dots}$

↳ for non-symmetric fuzzy sets (trapezoidal or triangular)

⇒ add defining points of fuzzy set & divide by their number

---

* Neural network [information processing paradigm]

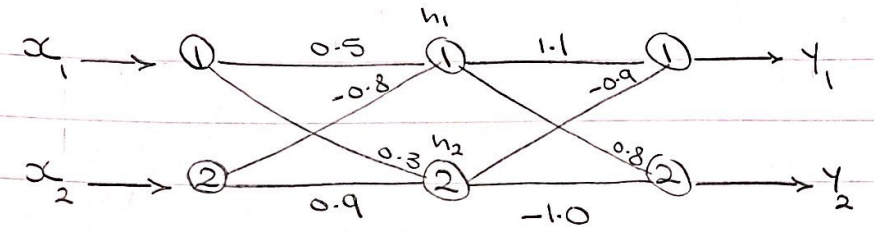
- ANN incorporate the two fundamental components of biological neural networks:
  (1) neurons (nodes)
  (2) Synapse (weights)
- Neural networks learn from examples [during taining period]
- Different types of neural networks:
  (1) feed-forward neural network:
  - one way only
  - no feedback
  (2) Feedback neural network
  - travel in both directions
  - their 'state' is changing continuously
- phases and data of an ANN:
  (1) Learning (training)
  - done by back-propagation
  - done using labled data
  - time consuming
  (2) Testing
  (3) Execution
  - feed forward is used
  - new unlabled data

Scanned with CamScanner

$\eta = 0.5$

Labled data

x_1	x_2	y_1	y_2
1	3	0.9	0.1



(1) Feed forward path.

- $h_1 in = 1 \times 0.5 + 3 \times (-0.8) = -1.9$

  $h_1 out = \frac{1}{1+e^{-1.9}} = 0.13$

- $h_2 in = 1 \times 0.3 + 3 \times 0.9 = 3.0$

  $h_2 out = \frac{1}{1+e^{-3.0}} = 0.95$

- $y_1 in = 0.13 \times 1.1 + 0.95 \times (-0.9) = -0.712$

  $y_1 out = \frac{1}{1+e^{-0.712}} = 0.329$

- $y_2 in = 0.13 \times 0.8 + 0.95 \times (-1.0) = -0.846$

  $y_2 out = \frac{1}{1+e^{-(-0.846)}} = 0.3$

(2) Calculate error of output layer:

- $Error\ y_1 = out_{y_1} (1 - out_{y_1}) (Target_{y_1} - out_{y_1})$
  $= 0.329\ (1 - 0.329)\ (0.9 - 0.329) = 0.126$
- $Error\ y_2 = out_{y_2} (1 - out_{y_2}) (Target_{y_2} - out_{y_2})$
  $= 0.3\ (1 - 0.3)\ (0.1 - 0.3) = -0.042$

(3) Change output layer weights:

- $w_{h_1 y_1}^{(t+1)} = w_{h_1 y_1}^{(t)} + \eta \times Error\ y_1 \times h_1 out$
  $= 1.1 + [0.5 \times 0.126 \times 0.13] = 1.10819$
- $w_{h_2 y_1}^{(t+1)} = w_{h_2 y_1}^{(t)} + \eta \times Error\ y_1 \times h_2 out$
  $= -0.9 + [0.5 \times 0.126 \times 0.95] = -0.84$
- $w_{h_1 y_2} = w_{h_1 y_2}^{(t)} + \eta \times Error\ y_2 \times h_1 out$
  $= 0.8 + [0.5 \times -0.042 \times 0.13] = 0.797$
- $w_{h_2 y_2} = w_{h_2 y_2}^{(t)} + \eta \times Error\ y_2 \times h_2 out$
  $= -1.0 + [0.5 \times -0.042 \times 0.95] = -1.019$

Scanned with CamScanner

(4) calculate (back-propagate) hidden layer error:

- $Error_{h_1} = out_{h_1}(1-out_{h_1}) \times (Error_{y_1} \times w_{h_1 y_1} + Error_{y_2} \times w_{h_1 y_2})$
  $= 0.13\,(1-0.13) \times (0.126 \times 1.1 + (-0.042) \times 0.8) = 0.01188$
- $Error_{h_2} = out_{h_2}(1-out_{h_2}) \times (Error_{y_1} \times w_{h_2 y_1} + Error_{y_2} \times w_{h_2 y_2})$
  $= 0.95\,(1-0.95) \times (0.126 \times (-0.9) + (-0.042) \times (-1.0) = -0.00339$

(5) change hidden layer weights

- $w_{x_1 h_1}^{(t+1)} = w_{x_1 h_1}^{(t)} + u \times error_{h_1} \times x_1$
  $= 0.5 + [0.5 \times 0.01188 \times 1] = 0.5059$
- $w_{x_2 h_1}^{(t+1)} = w_{x_2 h_1}^{(t)} + u \times error_{h_1} \times x_2$
  $= 0.8 + [0.5 \times 0.01188 \times 3] = -0.78$
- $w_{x_1 h_2}^{(t+1)} = w_{x_1 h_2}^{(t)} + u \times error_{h_2} \times x_1$
  $= 0.3 + [0.5 \times (-0.00339) \times 1] = 0.298$
- $w_{x_2 h_2}^{(t+1)} = w_{x_2 h_2}^{(t)} + u \times error_{h_2} \times x_2$
  $= 0.9 + [0.5 \times (-0.00339) \times 3] = 0.8949$

(6) Next feed forward path:

- $h_1 in = 1 \times 0.5059 + 3 \times -0.78 = -1.83$
- $h_1 out = 0.138$
- $h_2 in = 1 \times 0.298 + 3 \times 0.8949 = 2.98$
- $h_2 out = 0.95$
- $y_1 in = 0.138 \times 1.10819 + 0.951 \times (-0.84) = -0.6$
- $y_1 out = 0.344$
- $y_2 in = -0.85$
- $y_2 out = 0.297$

y_1 [0.9]

0.329 → 0.344

y_2 [0.1]

0.3 → 0.297

Scanned with CamScanner

* Back propagation:

Step 1: feed forward path

Step 2: Mean square error

$$Ep = \frac{1}{2} \sum_{k=1}^{n} (Target_k - output_k)^2$$

if Ep <= acceptable value then stop

Step 3: Calculate error of output layer

$$Error_\alpha = out_\alpha (1 - out_\alpha)(Target_\alpha - out_\alpha)$$

Step 4: Change output layer weights

$$W^+_{A\alpha} = W_{A\alpha} + n * Error_\alpha * out_A$$

Step 5: Calculate (back-propagate) hidden layer error

$$Error_A = out_A (1 - out_A) \times (Error_\alpha \times W_{A\alpha} + Error_\beta \times W_{AB})$$

Step 6: Change hidden layer weights

$$W^+_{\lambda A} = W_{\lambda A} + n * Error_A * in_\lambda$$

Step 7: Next feed forward path

Node biases: ⟶ output = sum (weights × inputs) + bias

- Bias allows you to shift the activation function by adding a constant to the input.

Network training:

(1) Supervised training:
⇒ inputs & desired outputs
⇒ weights are modified to reduce the difference between the actual & desired output

(2) unsupervised training:
⇒ inputs only

Learning rate: [η]

- Too small
  ↳ slow [so many iterations/takes abt of time]
- Too large
  ↳ not generalized / fast learning

Hidden layers & neurons:

↳ one layer is sufficient, two if function is discontinuous

neurons

- Too few
  ↳ NN can't learn the details
- Too many:
  ↳ NN learns the insignificant details

- **common error:** is to test the neural network using the same samples that were used in training
- **Training set:** a group of samples used to train the neural network
- **Testing set:** a group of samples used to test the performance of the neural network

Scanned with CamScanner

- Good training set can avoid overfitting

- What is a good example of training set?

  ↳ must represent the general population

  must contain members of each class (must contain wide range of variations or noise effect)

- Training FFNN using genetic algorithms

  * Each chromosome will represent the set of all weights of the network [fixed length]
  * Each gene represent 1 weight value
  * chromosome encoding is floating point
  * Random initialization
  * fitness : mean - square error is used
  * crossover : Single / multiple point
  * Mutation : Non-uniform floating point
  * Selection : Roulette wheel selection
  * Replacement : Elitism strategy

- Evolving fuzzy systems using genetic algorithms

* Each chromosome will represent a single rule set.
* Random initialization of the population.
* fitness : mean - square error is used

  note: the number of rules in chromosome is variable
* Selection : Roulette wheel selection
* Replacement. Elitism strategy

Scanned with CamScanner