Fuzzy Logic & Data Processing

Lecture notes for Modern Method of Data Processing (CCOD) in 2012

Akira Imada Brest State Technical University, Belarus

(last modified on)

December 11, 2012

PART I

Fuzzy Set Arithmetics

1 Fuzzy Set Theory

1.1 Fuzzy set vs. Crisp set

- Examples of crisp set
 - $\star 0 < x < 10$
 - $\star x=12$
- Examples of fuzzy set
 - \star {x is much smaller than 10}
 - \star {x is close to 12}
 - * Beer is either of {very-cold, cold, not-so-cold, warm}

1.1.1 Membership function

How x is likely to be A is expressed by a function called membership function. Usually it is described as $\mu_A(x)$.

For example, a possible membership function for a fuzzy expression $\{x \text{ is close to } 12\}$ will be

$$\mu(x) = \frac{1}{1 + (x - 12)^2} \tag{1}$$

See Figure 1.

1.1.2 AND and OR in Fuzzy Logic

In the logic of crisp set A and B and A or B are defined as in Figure 3.

In Fuzzy Logic, on the other hand, the membership function of A and B and A or B are specified in various way, but most popular ones are:

$$\mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}\tag{2}$$

and

$$\mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\},$$
 (3)

respectively.

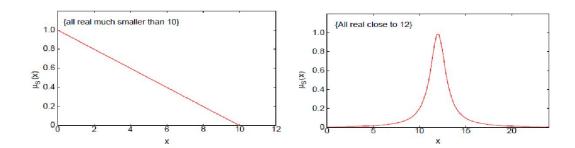


Figure 1: Examples of membership function $\{x \text{ is much smaller than } 10\}$ (right) and $\{x \text{ is close to } 12\}$ (left).

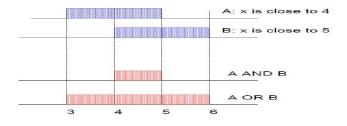


Figure 2: AND and OR in crisp set.

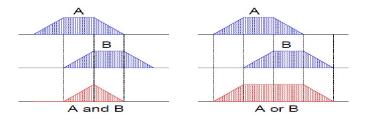


Figure 3: AND and OR in fuzzy set.

4

To be more concrete the membership function of x is closer to 4 AND/OR x is closer to 5 is like a Figure 4.

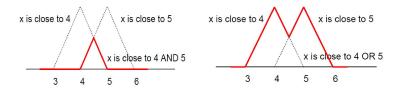
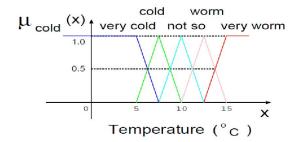


Figure 4: Membership function of x is closer to 4 OR x is closer to 5

• Very cold or pretty cold beer. ($\mu(x)$ is defined on temperature). Assume we like very cold beer or pretty cold beer and now we have a beer the temperature of which is 3 degree. Then how is the beer likely to be our prefered one?



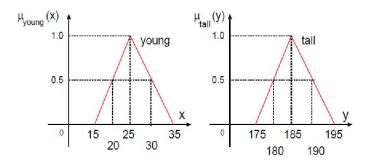
Note that this operation of OR was possible because both of the two membership function is defined on the same domain temperature. Then what if two membership functions are defined on different domains, such as age and height?

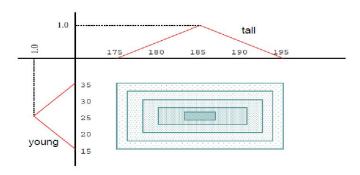
• Young and tall.

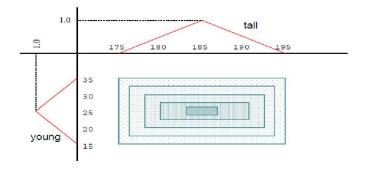
For example,

We cannot draw the membership function of young and tall on the 2-dimensional coordinate any more.

- (1) 3-D graphic ($z = \mu$ is defined on x = age and y = height)
- (2) Matrix representation koko







height \ age	0	10	20	30	40	50	60	70	80	90	100
150											
160											
170											
180											
190											

1.1.3 IF-Then rule in Fuzzy Logic

In Fuzzy Logic, the membership of $\mathit{IF}\ A$ $\mathit{Then}\ B$ is specified also in many way. Here, let's take it as

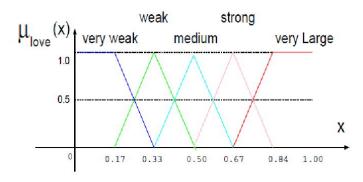
* Mamdani's proposal

$$\mu_{A\to B}(x) = \min\{\mu_A(x), \mu_B(x)\}\tag{4}$$

★ Larsen's proposal

$$\mu_{A \to B}(x) = \mu_A(x) \times \mu_B(x) \tag{5}$$

• If he is young then my love to him is strong.



• If he is young and tall then my love to him is very strong.

1.2 How to express multidimensional membershipfunction

PART II

Fuzzy Controller

2 Fuzzy Controller

Let's construct a virtual metro and control trains by fuzzy controller.

A goal

We now assume x is speed of my car, y is distance to the car in front, and z is how strongly we push brake-pedal. Then let's controll my car with a set of rules, like

- IF x is high and y is short THEN z should be strong
- IF x is medium and y is long THEN z should be medium
- IF x is medium low or x is medium and y is long THEN z should be weak
- IF x is low or x is medium low and y is short or y is medium hort THEN z should be medium weak
- etc.

Then the results will be plotted like in the Figure below.

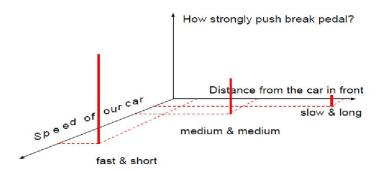


Figure 5: An example of the goal of Fuzzy controller

2.1 Virtual metro system with two trains in a loop line

We study Fuzzy Controllar via a simulation of virutal metro with one loop line on which two Train A and B run. To simplify we don't assume stations. That is, both trains always run. The speed of these trains are denoted as x_A and X_B . The distance from train A to train B is denoted as y_A and from train B to train A is y_B . Note that $x_A + y_B$ is constant (length of the loop line). Speed will be controlled by the distance to the train in front via its break. The shorter the distance, the storonger the break in order to avoid a collision.

Exercise 1 Create your own simulation of metro with one loop on which two trains A and B run, using graphics. 6 parameters x_A , x_B , y_A , y_B , z_A , z_B , should also be desplayed on the screen. The simulation might be started with $x_A = x_B$, $y_A = y_B$, $z_A = z_B = 0$.

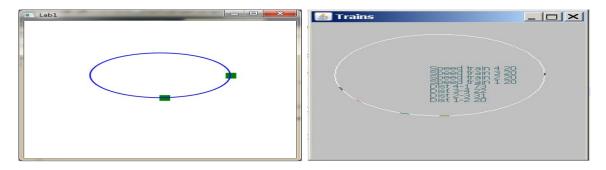


Figure 6: An example of a metro line. (Left) a standard implementation of a loop-line with two trains by Moroz Andrey. (Right) Four trains by Dvornichenko Valeriy.

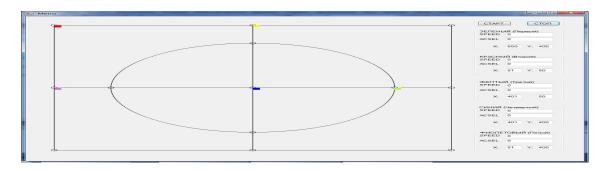


Figure 7: An example of a more complicated metro lines by Slusareva Maria (2012).

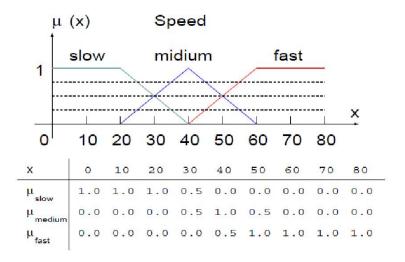


Figure 8: Membership function of x.

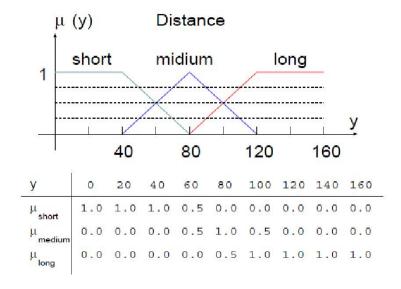


Figure 9: Membership function of y.

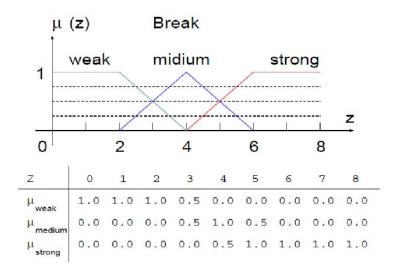


Figure 10: Membership function of z.

2.2 Let's design a set of rules for driving a train

2.2.1 An example of set of 9 rules to control break strength

IF X=FAST AND Y=SHORT THEN Z=STRONG
OR
IF X=FAST AND Y=MEDIUM THEN Z=WEAK
OR
IF X=MEDIUM AND Y=SHORT THEN Z=MEDIUM
OR
IF X=MEDIUM AND Y=MEDIUM THEN Z=STRONG
OR
IF X=SLOW AND Y=SHORT THEN Z=STRONG
...
etc.

2.3 How to construct a membership function of each rule

2.4 Membership function of Z under all possible pair of (X and Y)

Let's assume our rules are

Then the membership function of these two rules is

$$\max\{\min\{\mu_{medium}(x)\cdot\mu_{medium}(y),\mu_{medium}(z)\},\min\{\mu_{fast}(x)\cdot\mu_{short}(y),\mu_{strong}(z)\}\}$$

For example, let's calculate when x = 45 and y = 70

Z	1	2	3	4	5	6	7	8	9
$\mu(z)$	0.0000	0.0000	0.5000	0.5625	0.5000	0.5625	0.5625	0.5625	0.5625

Thus if every $\mu(z)$ is calculated for all the possible combination of x and y, we can draw a 3D plot such as

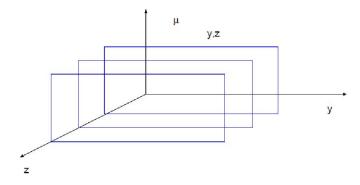


Figure 11: An example of the goal of Fuzzy controller

2.5 Defuzzification

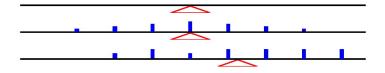


Figure 12: An example of the goal of Fuzzy controller

PART III

Fuzzy Data Mining

3 What is data mining

4 Classify data by a rule set

Assume we classify M data to be classified by using N features.

$$x_1, x_2, x_3, \cdots, x_N$$
.

A rule such as

IF
$$x_1 = A_1$$
 AND $x_2 = A_2$, AND \cdots , AND $x_N = A_N$ THEN class is ω_p .

classifies the data to one class ω_p .

 A_i is called attribute. For instance, (i) IF $x_i = 30$, (ii) IF $15 < x_i < 20$, (iii) IF x_i is Large, or (iv) IF x_i is Female, etc. The first two are called *crisp*, second is *fuzzy*, and fourth is called *categorical*. Let's take an example.

IF $x_1 = 20g$ AND $10cm < x_2 < 20cm$ AND $x_3 = Green$, AND $x_4 = Fruits$ THEN this is apple.

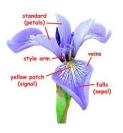
5 A benchmark – Iris database

As an example target here, we classify Iris flowers. Iris flower dataset¹ is made up of 150 samples consists of three species of iris flower, that is, setosa, versicolor and virginica. Each of these three families includes 50 samples. Each sample is a four-dimensional vector representing four attributes of the iris flower, that is, sepal-length, sepal-width, petal-length, and petal-width. All data are given as crisp as below.

Our mission is to train our classifiers with known data shown in Appendix as "data for training" and then evaluate how good is the classifiers with "data for checking" also shown in appendix.

Evaluation is how the system appropriately classifies 69 data. Ask system which family by giving 69 data one by one. Score is incremented if the result is correct. Hence the

 $^{^1{\}rm University}$ of California Urvine Machine Learning Repository. ics.uci.edu: pub/machine-learning-databases.



$\overline{x_1}$	x_2	x_3	x_4	class
0.65	0.80	0.20	0.08	1 (Setosa)
0.62	0.68	0.20	0.08	1 (Setosa)
0.59	0.73	0.19	0.08	1 (Setosa)
0.89	0.73	0.68	0.56	2 (Versicolor)
0.81	0.73	0.65	0.60	2 (Versicolor)
0.87	0.70	0.71	0.60	2 (Versicolor)
0.80	0.75	0.87	1.00	3 (Virginica)
0.73	0.61	0.74	0.76	3 (Virginica)
0.90	0.68	0.86	0.84	3 (Virginica)

maximum score is 69 and minimum is 0. Please note that even random guessing would score one out of three. So score 23 might be the most stupid classifier.

What you should show me are (1) run the algorithm and result of input x1, x2, x3, and x4 (2) a rule set, and (3) success rate (true-positive, true-negative, false-positive, false-negative).

For example

$$\text{IF}0.81 \leq x_1 \leq 0.89 \text{ AND } 0.70 \leq x_1 \leq 0.73 \text{ AND } 0.68 \leq x_1 \leq 0.71 \text{ AND } 0.56 \leq x_1 \leq 0.60x$$

THEN this is Versicolor

classifies $x_1 = 0.89$, $x_2 = 0.73$, $x_3 = 0.68$, $x_4 = 0.56$ properly to Versicolor while $x_1 = 0.65$, $x_2 = 0.80$, $x_3 = 0.20$, $x_4 = 0.08$ and $x_1 = 0.80$, $x_2 = 0.75$, $x_3 = 0.87$, $x_4 = 1.00$ are not.

So far so good, but what if the region overlaps with each other between more than two spieces? Or, what if unknown somewhat irregular data are given?

5.0.1 Evaluation of how good is a rule

The rule for class 1 should accept all data of class 1, but at the same time this rule should reject all data of class 2 and class 3. So count (i) how many data from class 1 are

14

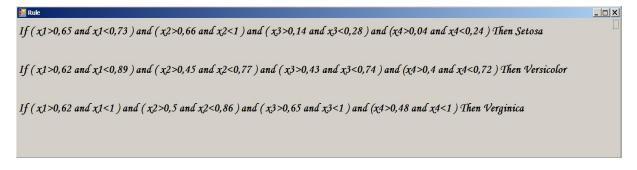


Figure 13: An example of a set of the simplest four rules. By Slusareva Maria (2012).

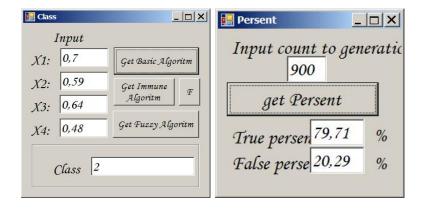


Figure 14: Example of applying a test data to the rule and its statistics. By Slusareva Maria (2012).

successfully accepted, and then (ii) how many data from class 2 are successfully rejected and (iii) how many data from class 2 are successfully rejected.

5.1 By Genetic Algorithm

Let's the rule is like

IF
$$a_1 < x_1 < a_1 + \delta_1$$
 AND $a_2 < x_2 < a_2 + \delta_2$ AND $a_3 < x_3 < a_3 + \delta_3$ AND $a_4 < x_4 < a_4 + \delta_4$ THEN class is p .

Then our chromosome will be like

$$(a_1 \ \delta_1 \ a_2 \ \delta_2 \ a_3 \ \delta_3 \ a_4 \ \delta_4)$$

where if $a_i + \delta_i > 1$ then we replace the value with 1. For example,

$$(0.23\ 0.41\ 0.52\ 0.81\ 0.89\ 1.00\ 0.11\ 0.72)$$

Create 40 such a chromosome at random which construct a population of the first generation. Then evolve the population with uniform chrossover and mutation under trancate selection with the target being the family 1. Then repeat this with family 2 and 3.

Probably one rule for one family is not enough. Therefore a Multi Objective Optimization will be necessary.

Algorithm 1 (1) For p=1 to 4

- (2) generate a population of 40 chromosomes at random
- (3) select two parents p_1 and p_2 at random.
- (4) create child c by uniform crossover and mutation
- (5) If $distance(c, p_1) < distance(c, p_2)$ and $fitness(c \dot{c} fitness(p_1))$ Then $replace p_1$ with c
- (6) ElseIf $distance(c, p_1) > distance(c, p_2)$ and $fitness(c \dot{c} fitness(p_2))$ Then $replace p_2$ with c
- (7) Repeat (3)-(6) until all the chromosom is replaced
- (8) Repeat (3)-(7) until all the chromosom never be changed
- (9) End For

5.1.1 Fitness

Fitness might be evaluated by

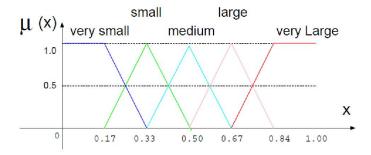
$$f = \frac{TP}{TP + FN} \times \frac{TN}{FP + TN} \tag{6}$$

where TP stands for true-positive, FN stands for false-negative, TN stands for true-negative, and FP stands for false-positive, and true-positive is the number of cases covered by the rule that have the class predicted by the rule; false-positives is the number of cases covered by the rule that have a class different from the class predicted by the rule; false-negatives is the number of cases that are not covered by the rule but that have the class predicted by the rule; true-negatives is the number of cases that are not covered by the rule and do not have the class predicted by the rule.

5.1.2 Result to be shown

	Setosa		Versi	icolor	Virg	inica
	yes	no	yes	no	yes	no
rule-01 for class 1	100	0	0	100	0	100
rule- 02 for class 1	100	0	0	100	0	100
rule- 03 for class 1	100	0	0	100	0	100
rule-04 for class 1	100	0	0	100	0	100
rule-05 for class 2	0	100	100	0	0	100
rule-06 for class 2	0	100	100	0	0	100
rule-07 for class 2	0	100	100	0	0	100
rule-08 for class 2	0	100	100	0	0	100
rule-09 for class 3	0	100	0	100	100	0
rule-10 for class 3	0	100	0	100	100	0
rule-11 for class 3	0	100	0	100	100	0
rule-12 for class 3	0	100	0	100	100	0

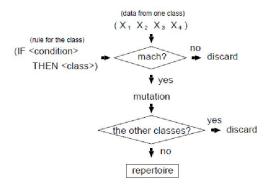
5.2 Fuzzy approach



5.3 Immune approach 1 - exhaustive search

Algorithm 2 (1) For $\omega = 1$ to 3

- (2) For i = 1 to 40
- (3) generate a rule
- (4) if the rule properly predict the family keep it in the repertoire otherwise discard.
- (5) repeat (2) to (4)
- (6) repeat (1) to (5)



5.4 Immune approace 2 - Clonal Selection

This algorithm is proposed by ... for the purpose of pattern classification being based on the algorithm called CLONALG proposed by ... for the purpose of pattern recognition.

Algorithm 3 Artificial Cronal Selection algorithm

Assume we have a set of Antigens Ag^2 .

- 1. Generate an initial population of N antibodies Ab
- 2. Select an antigen Ag_i
- 3. For every member of Ab calculate its affinity to the Ag_i
- 4. Select the n highest affinity antibodies and generate a number of clones for each antibody in proportion to its affinity and construct a new population C^i (a population specifically for Ag_i)
- 5. Mutate every member of C^i with a probability being inversely proportional to its affinity. Let's call this population a mature population C^{i*}
- 6. Measure again affinity of the member C^{i*} to the Ag_i . (If its affinity is greater than the current memory cell Abm_i it become the new memory cell.
- 7. (a) Replace p highest affinity antibodies in Ab with p highest affinity matured antibodies in C^{i*}
 - (b) Replace (N-p) low affinity antibodies in Ab with a new set of (N-p) random antibodies.
- 8. Repeat 2-7 for all antigens Ag_i (i=1, 2, ..., M)
- 9. Repeat 1-8 for all different antigens³.

Thus we can create N mature antibody population C^{i*} for each of antigen Ag_i . And one memory cell for each antigen.

²Here we take each of three different iris flowers as antigen, that is, we have three Antigens called Setosa, Versicolor, and Virsinica.

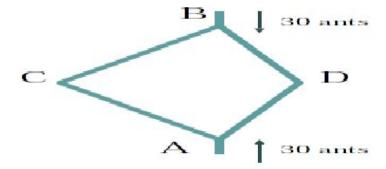
³That is, Setosa, Versicolor, and Virsinica.

5.5 Ant Colony Optimization (ACO) Approach

This subsection is edited from G. Sarigiannidis⁴.

5.5.1 ACO in general





Which path the ants should choose

Each ant who is at the city s choose the next city j from J(s) – its not-yet-visited-city from city s – with a probability:

$$p_{sj} \propto \tau_{sj}(t) \{\eta(s,j)\}^{\beta}.$$

Or, to normalize it:

$$p_{sj} = \frac{\tau_{sj}(t) \{ \eta(s,j) \}^{\beta}}{\sum_{j \in J(s)} \tau_{ij}(t) \{ \eta(s,j) \}^{\beta}}$$

where $\tau_{ij}(t)$ is the amount of pheromone on the path from city s to city j, $\eta(s,j) = 1/d_{sj}$ while d_{sj} is the distance between city s and city j, and β is a parameter which determines the importance of balance between pheromone and distance.

⁴G. Sarigiannidis "Discussion on the implementation of the Ant Colony System for the Traveling Salesman Problem." Delft University of Technology

Local pheromon update

When all the ants reach the next city, the amount of pheromon is updated as:

$$\tau_{rs} = (1 - \rho) \cdot \tau_{rs} + \rho \cdot \tau_0$$

Global pheromon update

After all ants have complete their tours, they remain to the position for the next iteration, and the global pheromone update is applied to all the path (r, s) one by one as follows:

$$\tau_{rs} = (1 - \alpha) \cdot \tau_{rs} + \alpha \cdot \Delta \tau_{rs}$$

where $\Delta \tau_{rs} = 1/(L_{\rm BS})$ if the path (r,s) belongs to the best tour so far, otherwise 0 where $L_{\rm BS}$ is the minimum total tour length found so far. An interpretation of $0 < \alpha < 1$ can be the pheromone decay parameter, that is, evaporation rate of pheromone.

Parameters

Typically, the number of ants is 10, and the other parameters are $\beta = 2$, $\alpha = \rho = 0.1$, $\tau_0 = 1/(n \cdot L_{\text{NN}})$ where L_{NN} is the total tour length by Nearest Neighbor heuristic and n is a number of cities.

The ant who is in the node i chooses the next node to move from a set of possible nodes directly reachable with a probability p_{ij} which are pre-specified in advance a run constant during a run, by considering (i) η_{ij} – how the path to the node j looks good (called heuristic term) and (ii) τ_{ij} – how the path accumulate pheromon those previous ants left.

5.5.2 Traveling Salesperson Problem by ACO

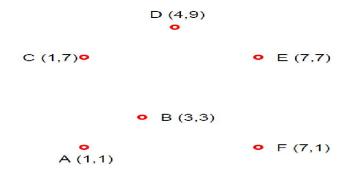
In order to understand how Ant Colony Optimization works, let's take a brief look at its application to the Traveling Salesperson Problem. We now assume N cities that a salesperson must visit once and only once. Now imagine N ants are assigned to each of those N cities – one ant at one city. The following parameters are to be used.

 \star probability for one ant at node i to choose node j

$$p_{ij} \propto \eta_{ij} \tau_{ij}^{\beta}$$

where $\beta(>0)$ determines the relative importance of pheromone versus heuristics, and η is called *heuristic parameter*. The larger value, the more preferable. In the TSP η_{ij} simply takes a value $1/d_{ij}$ where d_{ij} is the distance between city i and city-j. τ is called *pheromone density* and updated during a run in the following way.

Try to apply the following 6 cities TSP.



5.5.3 Data Classification by ACO

We use Iris-flower data again and the rules to classify iris-flower data are described with fuzzy linguistic term. A general form of such rule is:

IF x_1 is small & x_2 is large & \cdots THEN y_1 is strong & y_2 is weak & \cdots

Here, we assume those linguistic terms such as *small*, *large*, etc for x_i , and *strong*, *weak*, etc. for y_j are all pre-defiend by a membership function.

Remember here how we define, for example, a membership function of

Then we can define a membership function of these rules.

$$\mu_{A\cup B\Rightarrow C\cup D}$$
.

Our purpose here is to classify iris-flower data with four features into either of three families. So the form of our rule is specifically like:

IF
$$x_1$$
 is $A_1 \& x_2$ is $A_2 \& x_3$ is $A_3 \& x_4$ is A_4 THEN y is B_i

where A_i is either of small, medium, or large and B_j is either of 1, 2, or 3.

Then the number of all possible such rules is $3^4 \times 3 = 243$. Now let's denote IF-part of the *i*-th roule as R_i . Now we assign heuristic term for each of those links from A_i to B_j as

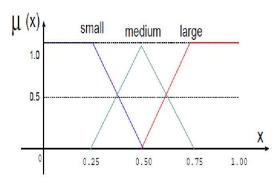


Figure 15: Three membership function of "small", "medium", "large" for all x_1 , x_2 , x_3 and x_4 in commone.

$$\eta_{ij} = \mu_{R_i \Rightarrow B_i}(x_1, x_2, x_3, x_4, y_1) \tag{7}$$

For example

$$\eta_{ij} = \max \min(\mu_{A_1}(x_1), \mu_{A_2}(x_2), \mu_{A_3}(x_3), \mu_{A_4}(x_4))$$

See Figure 17.

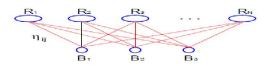


Figure 16: A schematic diagram of all possible link between IF-part and THEN-part of rule set. Heuristic term η_{ij} are assingend to those links.

With initial pheromone being arbitrary τ_0 , pheromon on the link from i to j is updated every time when one ant walks this link (local update) as

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \rho \cdot \tau_0. \tag{8}$$

When one ant finish all the R_i pheromon on all the like are updated (global update) as

$$\tau_{ij} = (1 - \rho) \cdot \tau_{ij} + \rho \cdot \Delta_{ij} \tag{9}$$

where Δ_{ij} is fitness of the best rule so far found if the link from i to j belongs to the link of the best rule so far found, and 0 otherwise. Use Eq. (6) for fitness calculation of one rule. The calculation is made by using the data for training. Note that it is this process

	X 1	X 1	X 1	X 1	В
R 1:	small	small	small	small	1
R 2:	small	small	small	small	2
R3:	small	small	small	small	3
R 4:	small	small	small	medium	1
R 5:	small	small	small	medium	2
R 6:	small	small	small	medium	3
R 7:	small	small	small	large	1
					• • •
R ₈₁ :	large	large	large	large	3

Figure 17: All possible rules are something like this.

that the system learn how to classify well.

The probability of one ant at the node i to choose the link to the node k is as already described:

$$p_{ij} = \frac{\tau_{ij}(t) \{ \eta(i,j) \}^{\beta}}{\sum_{j \in J(i)} \tau_{ij}(t) \{ \eta(i,j) \}^{\beta}}$$

with β being, say, 2.

Now let's denote input-output data set from our 120 training data of iris-flowers as

$$\mathbf{e}^k = (x_1^k, x_2^k, x_3^k, x_4^k, y^k)$$
 $k = 1, 2, \dots, 120.$

Then our rules are like

 \mathbb{R}^s : IF x_1 is large & x_2 is small & x_3 is medium & x_4 is small THEN 2.

To be more general

$$R^s$$
: IF x_1 is A_1^s & x_2 is A_2^s & x_3 is A_3^s & x_4 is A_4^s THEN B^s .

Then

$$\eta_{ij} = \max_{e^k} (\min(\mu_{A^i}(\mathbf{x}^k), \mu_{B^j}(y^k)))$$
(10)

where

$$\mu_{A^i}(\mathbf{x}^k) = \min(\mu_{A_1^i}(x_1^k), \mu_{A_2^i}(x_2^k), \mu_{A_3^i}(x_3^k), \mu_{A_4^i}(x_4^k))$$
(11)

Let me take an example. Assume s = 1 and

 \mathbb{R}^1 : IF x_1 is small & x_2 is small & x_3 is small & x_4 is small THEN 1.

That is,

 A_1^1 is small, A_2^1 is small, A_3^1 is small and A_4^1 is small.

Then from the iris-flower data for training for k=1

$$x_1 = 0.65$$
, $x_2 = 0.80$, $x_3 = 0.20$, $x_4 = 0.08$ and $y = 1$.

That is,

$$e^1 = (0.65, 0.80, 0.20, 0.08, 1).$$

Corresponding values of membership function

$$\mu_{small}(0.65), \, \mu_{small}(0.80), \, \mu_{small}(0.20), \, \mu_{small}(0.08)$$

are obtained from Fig. ??.

By repeating this procedure from k = 1 to k = 120, and then take the maximum in Eq.(10) we can obtain each value of η_{ij} .

As for B^s , they are generally also a fuzzy linguistic value such as strong and its membership function is defined in the same way as Fig. ??, but here B^s is integer value 1, 2, or 3, that is singleton. Therefore, for example $\mu_{setosa}(y) = 1$ if and only if y=1 otherwise $\mu_{setosa}(y) = 0$. Similarly, $\mu_{versicolor}(y) = 1$ if and only if y=2 otherwise $\mu_{versicolor}(y) = 0$, and $\mu_{virsinica}(y) = 1$ if and only if y=3 otherwise $\mu_{virginica}(y) = 0$.

Now we are ready to run an ACO algorithm for Iris-flower classification

Algorithm 4 (ACO for iris-flower classification) Initialize τ_0 , calculate all η_{ij}

- 1. For k = 1 to K.
- 2. For i = 1 to 243.
- 3. Put ant k to the node R_i .
- 4. Choose link according to p_{ij} updating t_{ij} locally.
- 5. Set THEN-part of the " R_i as B_j .
- 6. Repeat 2-5.
- 7. Update all of the τ_{ij} globally according to the fitness of the rule obtained..
- 8. Repeat 1-7.

PART III

Fuzzy Neural Network with Takagi-Sugeno Model

5.6 A Fuzzy Neural Network Approach

The goal is to classify the data taken from the n-dimensional data-set into either of the pre-defined m classes. For the purpose, Castellano et al. [?] used the inference mechanism of the zero-order Takagi-Sugeno fuzzy model; then realized the idea by a fuzzy neural network model. To train the fuzzy neuronal network, they employed a combination of (i) a competitive learning to determine the architecture of the fuzzy neural network at first and (ii) a gradient descent learning to optimize the synaptic weights afterwards. We, on the other hand, employ an evolutionary computation technique to train the network, since we already know the optimal network structure under our current interest, and as such, our concern is just to obtain the solution of weight configuration of the fuzzy neural network.

In the following three sub-subsections, Takagi-Sugeno fuzzy model, a realization of the model by fuzzy neural network, and how we optimize the weight of the fuzzy neural network by an evolutionary computation are described more in detail.

5.6.1 Takagi-Sugeno Model.

Though Castellano et al. [?] stated the method very clearly in their paper, let us briefly describe it with an intention of making this paper self-contained. Takagi-Sugeno fuzzy inference model is made up of a set of H rules, such as

$$R_k$$
: IF $(x_1 \text{ is } A_1^k)$ and \cdots and $(x_n \text{ is } A_n^k)$
THEN $(y_1 \text{ is } \nu_{k1})$ and \cdots and $(y_m \text{ is } \nu_{km})$

where R_k is the k-th rule $(k = 1, \dots, H)$, x_i denotes the i-th variable of the input data $(i = 1, \dots, n)$, y_j denotes the j-th output variable $(j = 1, \dots, m)$, A_i^k denotes a fuzzy set which is usually expressed by a linguistic term such as "Medium-Large" but here expressed by a shape of membership function defined one by one on the corresponding input variable, and ν_{kj} denotes a fuzzy singleton each defined on the output variables indicating the likeliness of how the input belongs to the j-th class according to the k-th rule.

 A_i^k is defined by Gaussian membership functions

$$\mu_{ik}(x_i) = \exp\{-(x_i - w_{ik})^2 / \sigma_{ik}^2\}.$$

Then defuzzification for an input $\mathbf{x}^0 = (x_1^0, \dots, x_n^0)$ is via the equation:

$$y_j^0 = \{\sum_{k=1}^H (\mu_k(\mathbf{x}^0) \cdot \nu_{kj})\} / \sum_{k=1}^H \mu_k(\mathbf{x}^0)$$

where

$$\mu_k(\mathbf{x}^0) = \prod_{i=1}^n \mu_{ik}(x_i^0)$$

is the results of application of the Larsen product operator.

In other words, the procedure of inference is as follows. When an input $\mathbf{x} = (x_1, \dots x_n)$ is given, each of the H rules evaluates the \mathbf{x} and output the likeliness of the class, from one class to the next, to which \mathbf{x} belongs to. The evaluation by k-th rule of x_i is by the corresponding membership function $\mu_{ik}(x_i)$ which is specified by giving two parameters w_{ik} and σ_{ik} so that it returns a value ranging from 0 to 1. See, e.g., Fig. 1 where the i-th coordinate of the input \mathbf{x} is evaluated by A_i^k , the i-th antecedent of the IF part of the Rule_k, which is represented by a membership function not by a usual linguistic term like "Small". The returned membership value in this example in the figure is 0.71, suggesting, say, "The likeliness of if the variable is "Medium Large" is 0.71."

Figure 18: A fictitious sketch of an evaluation of x_i , the *i*-th entry of the input \mathbf{x} , by the *i*-th antecedent part of the *k*-th rule A_i^k .

Using those n values of $\mu_{ik}(x_i)$, each of the H rules calculates $\mu_k(\mathbf{x})$, and finally these H values are combined to calculate m values of y_j , the resultant defuzzified value for each of the m classes.

5.6.2 Fuzzy Neural Network Implementation.

The procedure described in the previous sub-subsection can be realized when we assume a neural network architecture such as depicted in Fig. 2. The 1st layer is made up of n input neurons. The 2nd layer is made up of H groups of a neuronal structure each contains n neurons where the i-th neuron of the k-th group has a connection to the i-th neuron in the 1st layer with a synaptic connection which has a pair of weights (w_{ik}, σ_{ik}) . Then k-th group in the second layer calculates the value $\mu_k(\mathbf{x})$ from the values which are received from each of the n neurons in the first layer. The 3rd layer is made up of m neurons each of which collects the H values from the output of the second layer, that is

j-th neuron of the 3rd layer receives the value from k-th output in the second layer with the synapse which has the weight ν_{kj}

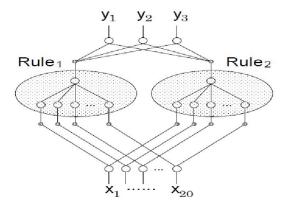


Figure 19: Architecture of the proposed fuzzy neural network which infers how an input $\mathbf{x} = (x_1, \dots x_n)$ is likely to belong to the *j*-th class by generating outputs y_j each of which reflect the degree of the likeliness. In this example, a 20-dimension data input will be inferred to which of the 3 classes the input belongs by using 2 rules.

5.6.3 How it learns?

Castellano et al. [?] used (i) a competitive learning to determine how many rules are needed under initial weights created at random. Then, in order to optimize the initial random weight configuration, they use (ii) a gradient method performing the steepest descent on a surface in the weight space employing the same training data, that is, supervised learning.

Here, on the other hand, we use a simple genetic algorithm, since our target space is specific enough to know the network structure in advance, i.e., only unique rule is necessary. Our concern, therefore, is just obtaining the solution of weight configuration of the network. That is to say, all we want to know is a set of parameters w_{ik} , σ_{ik} and ν_{kj} $(i=1,\cdots n)$, $(k=1,\cdots H)$, $(j=1,\cdots m)$ where n is the dimension of data, H is the number of rules, and m is the number of outputs. Hence our chromosome has those $n \times H \times m$ genes. Starting with a population of chromosomes whose genes are randomly created, they evolve under simple truncate selection where higher fitness chromosome are chosen, with uniform crossover and occasional mutation by replacing some of a few genes with randomly created other parameters, expecting higher fitness chromosomes will be emerged. These settings are determined by trials and errors experimentally.

PART IV

Fuzzy Relation

6 Fuzzy Relation

In this section we study fuzzy expressions such as "at least middle-aged," brighter than average," more or less expensive" and "younger than about 20."

First, let's recall Cartesian product $X \times Y$ in which both X and Y is a set. Let me take an example. Assume now $X = \{1,2\}$ and $Y = \{a,b,c\}$ then $X \times Y = \{(1,a),(1,b),(1,c),(2,a),(2,b),(2,c)\}$. Then relation is defined over Cartesian product $X \times Y$, that is, a subset of $X \times Y$. In other words relation is a set of ordered pair in which order is important.

Generally it is defined over multipel set, like $X_1 \times X_2 \times \cdots \times X_n$, but here we think of only product of two set, and call it binary relation.

To visualize we can plot $\mu_R(X,Y)$ 3-D Cartesian space.

* Example 1 ...
$$X = \{1, 2\}, Y = \{2, 3, 4\}, R : X < Y$$

Let's think of it as a *crisp* logic, that is, the value is 1 (yes) or 0 (no). Then membership function of this relation will be:

$X \setminus Y$	2	3	4
1	1	0	0
2	1	1	0
3	0	0	1

Then what about the relation $R: x \approx y$. Let's think of this example with fuzzy logic.

$$\star$$
Example 2 ... $X=\{1,2\},\,Y=\{2,3,4\},\,R:X\approx Y$

$X \setminus Y$	2	3	4
1	2/3	1/3	0
2	1	2/3	1/3
3	2/3	1	2/3

The values are just examples. Further more we think of X and Y as a continuous values instead of integer. Then membership function is a surface instead of just 9 points, over X - Y coordinate.

We now proceed to examples where we use fuzzy linguistic expression instead of numbers.

 \star Example 3 ... $X = \{green, yellow, red\}, Y = \{unripe, semiripe, ripe\}.$

Imagine an apple. First, with a *crisp* logic. A red apple is usually ripe but a green apple is unripe. Thus:

$X \setminus Y$	unripe	semiripe	ripe
green	1	0	0
yellow	0	1	0
red	0	0	1

Now, secondly, with a *fuzzy* logic. A red apple is *provably* ripe, but a green apple is *most* likely, and so on. Thus, for example:

$X \setminus Y$	unripe	semiripe	ripe
green	1	0.5	0
yellow	0.3	1	0.4
red	0	0.2	1

These matrices are not necessarily rectangular. For example:

 \star Example 4 ... $X = \{Brest, London, BuenosAires\}$ Y=Tokyo, NewYork, Minsk, Johanesburg R: very far.

$\overline{X \setminus Y}$	Tokio	New York	Minsk	Johanesburg
Brest				
London				
Buenos Aires				

Try to fill those blancs by yourself.

APPENDIX

Iris Flower Database

Data for training - $3 \times 40 = 120$ data







	Set	osa			Versi	color			Virg	inica	
x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
0.65	0.80	0.20	0.08	0.89	0.73	0.68	0.56	0.80	0.75	0.87	1.00
0.62	0.68	0.20	0.08	0.81	0.73	0.65	0.60	0.73	0.61	0.74	0.76
0.59	0.73	0.19	0.08	0.87	0.70	0.71	0.60	0.90	0.68	0.86	0.84
0.58	0.70	0.22	0.08	0.70	0.52	0.58	0.52	0.80	0.66	0.81	0.72
0.63	0.82	0.20	0.08	0.82	0.64	0.67	0.60	0.82	0.68	0.84	0.88
0.68	0.89	0.25	0.16	0.72	0.64	0.65	0.52	0.96	0.68	0.96	0.84
0.58	0.77	0.20	0.12	0.80	0.75	0.68	0.64	0.62	0.57	0.65	0.68
0.63	0.77	0.22	0.08	0.62	0.55	0.48	0.40	0.92	0.66	0.91	0.72
0.56	0.66	0.20	0.08	0.84	0.66	0.67	0.52	0.85	0.57	0.84	0.72
0.62	0.70	0.22	0.04	0.66	0.61	0.57	0.56	0.91	0.82	0.88	1.00
0.68	0.84	0.22	0.08	0.63	0.45	0.51	0.40	0.82	0.73	0.74	0.80
0.61	0.77	0.23	0.08	0.75	0.68	0.61	0.60	0.81	0.61	0.77	0.76
0.61	0.68	0.20	0.04	0.76	0.50	0.58	0.40	0.86	0.68	0.80	0.84
0.54	0.68	0.16	0.04	0.77	0.66	0.68	0.56	0.72	0.57	0.72	0.80
0.73	0.91	0.17	0.08	0.71	0.66	0.52	0.52	0.73	0.64	0.74	0.96
0.72	1.00	0.22	0.16	0.85	0.70	0.64	0.56	0.81	0.73	0.77	0.92
0.68	0.89	0.19	0.16	0.71	0.68	0.65	0.60	0.82	0.68	0.80	0.72
0.65	0.80	0.20	0.12	0.73	0.61	0.59	0.40	0.97	0.86	0.97	0.88
0.72	0.86	0.25	0.12	0.78	0.50	0.65	0.60	0.97	0.59	1.00	0.92
0.65	0.86	0.22	0.12	0.71	0.57	0.57	0.44	0.76	0.50	0.72	0.60
0.68	0.77	0.25	0.08	0.75	0.73	0.70	0.72	0.87	0.73	0.83	0.92
0.65	0.84	0.22	0.16	0.77	0.64	0.58	0.52	0.71	0.64	0.71	0.80
0.58	0.82	0.14	0.08	0.80	0.57	0.71	0.60	0.97	0.64	0.97	0.80
0.65	0.75	0.25	0.20	0.77	0.64	0.68	0.48	0.80	0.61	0.71	0.72

(to be cont'd to the next page)

(cont'd)

	Set	osa			Versi	color			Virg	inica	
$\overline{x_1}$	x_2	x_3	x_4	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
0.61	0.77	0.28	0.08	0.81	0.66	0.62	0.52	0.85	0.75	0.83	0.84
0.63	0.68	0.23	0.08	0.84	0.68	0.64	0.56	0.91	0.73	0.87	0.72
0.63	0.77	0.23	0.16	0.86	0.64	0.70	0.56	0.78	0.64	0.70	0.72
0.66	0.80	0.22	0.08	0.85	0.68	0.72	0.68	0.77	0.68	0.71	0.72
0.66	0.77	0.20	0.08	0.76	0.66	0.65	0.60	0.81	0.64	0.81	0.84
0.59	0.73	0.23	0.08	0.72	0.59	0.51	0.40	0.91	0.68	0.84	0.64
0.61	0.70	0.23	0.08	0.70	0.55	0.55	0.44	0.94	0.64	0.88	0.76
0.68	0.77	0.22	0.16	0.70	0.55	0.54	0.40	1.00	0.86	0.93	0.80
0.66	0.93	0.22	0.04	0.73	0.61	0.57	0.48	0.81	0.64	0.81	0.88
0.70	0.95	0.20	0.08	0.76	0.61	0.74	0.64	0.80	0.64	0.74	0.60
0.62	0.70	0.22	0.04	0.68	0.68	0.65	0.60	0.77	0.59	0.81	0.56
0.63	0.73	0.17	0.08	0.76	0.77	0.65	0.64	0.97	0.68	0.88	0.92
0.70	0.80	0.19	0.08	0.85	0.70	0.68	0.60	0.80	0.77	0.81	0.96
0.62	0.70	0.22	0.04	0.80	0.52	0.64	0.52	0.81	0.70	0.80	0.72
0.56	0.68	0.19	0.08	0.71	0.68	0.59	0.52	0.76	0.68	0.70	0.72
0.65	0.77	0.22	0.08	0.70	0.57	0.58	0.52	0.87	0.70	0.78	0.84

Data for evaluating the system after training - $3 \times 23 = 69$ data



Setosa				Versicolor				Virginica			
$\overline{x_1}$	x_2	x_3	x_4	x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
0.63	0.80	0.19	0.12	0.70	0.59	0.64	0.48	0.85	0.70	0.81	0.96
0.57	0.52	0.19	0.12	0.77	0.68	0.67	0.56	0.87	0.70	0.74	0.92
0.56	0.73	0.19	0.08	0.73	0.59	0.58	0.48	0.73	0.61	0.74	0.76
0.63	0.80	0.23	0.24	0.63	0.52	0.48	0.40	0.86	0.73	0.86	0.92
0.65	0.86	0.28	0.16	0.71	0.61	0.61	0.52	0.85	0.75	0.83	1.00
0.61	0.68	0.20	0.12	0.72	0.68	0.61	0.48	0.85	0.68	0.75	0.92
0.65	0.86	0.23	0.08	0.72	0.66	0.61	0.52	0.80	0.57	0.72	0.76
0.58	0.73	0.20	0.08	0.78	0.66	0.62	0.52	0.82	0.68	0.75	0.80
0.67	0.84	0.22	0.08	0.65	0.57	0.43	0.44	0.78	0.77	0.78	0.92
0.63	0.75	0.20	0.08	0.72	0.64	0.59	0.52	0.75	0.68	0.74	0.72
0.62	0.70	0.22	0.04	0.80	0.52	0.64	0.52	0.81	0.70	0.80	0.72
0.56	0.68	0.19	0.08	0.71	0.68	0.59	0.52	0.76	0.68	0.70	0.72
0.65	0.77	0.22	0.08	0.70	0.57	0.58	0.52	0.87	0.70	0.78	0.84
0.63	0.80	0.19	0.12	0.70	0.59	0.64	0.48	0.85	0.70	0.81	0.96
0.57	0.52	0.19	0.12	0.77	0.68	0.67	0.56	0.87	0.70	0.74	0.92
0.56	0.73	0.19	0.08	0.73	0.59	0.58	0.48	0.73	0.61	0.74	0.76
0.63	0.80	0.23	0.24	0.63	0.52	0.48	0.40	0.86	0.73	0.86	0.92
0.65	0.86	0.28	0.16	0.71	0.61	0.61	0.52	0.85	0.75	0.83	1.00
0.61	0.68	0.20	0.12	0.72	0.68	0.61	0.48	0.85	0.68	0.75	0.92
0.65	0.86	0.23	0.08	0.72	0.66	0.61	0.52	0.80	0.57	0.72	0.76
0.58	0.73	0.20	0.08	0.78	0.66	0.62	0.52	0.82	0.68	0.75	0.80
0.67	0.84	0.22	0.08	0.65	0.57	0.43	0.44	0.78	0.77	0.78	0.92
0.63	0.75	0.20	0.08	0.72	0.64	0.59	0.52	0.75	0.68	0.74	0.72