Q.1 a. Define Union, Intersection and complement operations of Fuzzy sets.
Answer:
For fuzzy sets A and B

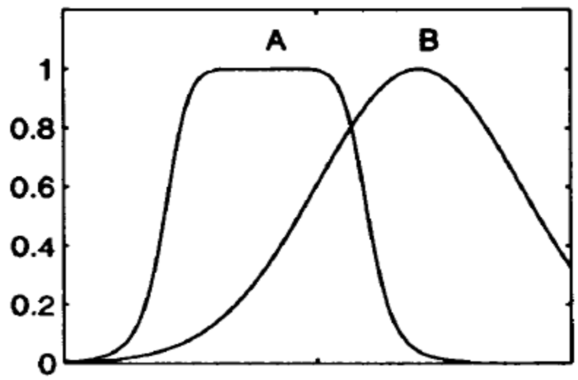


Figure Fuzzy sets A & B

The union of two fuzzy sets A and B is a fuzzy set C, written as C=AUB or C=A OR B, whose membership function (MF) is related to those of A and B by

$$\mu_c(x) = \max(\mu_A(x), \mu_B(x)) = \mu_A(x) \vee \mu_B(x)$$

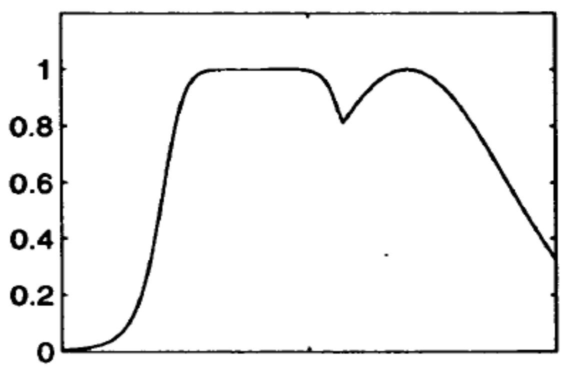


Figure C= A OR B

The intersection of two fuzzy sets A and B is a fuzzy set C, written as C=A∩B or C=A AND B, whose membership function (MF) is related to those of A and B by

$$\mu_C(x) = \min(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x)$$

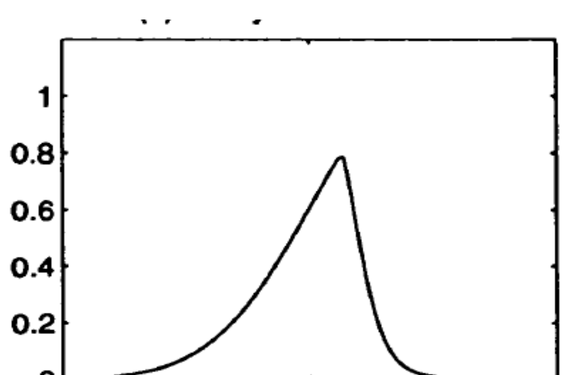


Figure C= A AND B

The complement of fuzzy set A, denoted by A', is defined as

$$\mu_{A'}(x) = 1 - \mu_A(x)$$

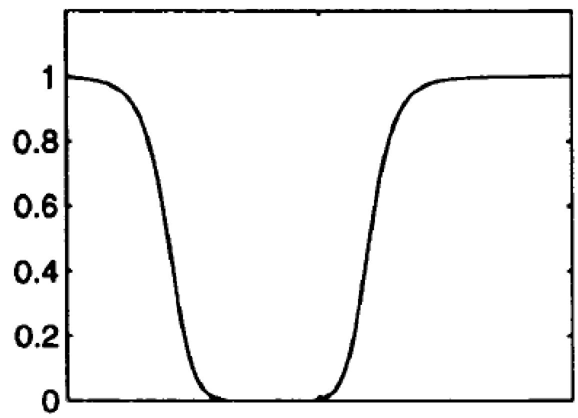


Figure C= A'

Q.1 b. Write Advantages of Sugeno and Mamdani methods.

Answer:

Advantages of the Sugeno Method

– It is computationally efficient.
– It works well with linear techniques (e.g., PID control).
– It works well with optimization and adaptive techniques.
– It has guaranteed continuity of the output surface.
– It is well suited to mathematical analysis.

Advantages of the Mamdani Method

– It is intuitive.
– It has widespread acceptance.
– It is well suited to human input.

Fuzzy inference system is the most important modeling tool based on fuzzy set theory. The FISs are built by domain experts and are used in automatic control, decision analysis, and various other expert systems

Q.1 c. Discuss common applications of Artificial Neural Network (ANN)

Answer:

Sol.) The tasks artificial neural networks are applied to tend to fall within the following broad categories:

- Function approximation, or regression analysis, including time series prediction, fitness approximation and modeling.
- Classification, including pattern and sequence recognition, novelty detection and sequential decision making.
- Data processing, including filtering, clustering, blind source separation and compression.
- Robotics, including directing manipulators, Computer numerical control.

Q.1 d. What is Attribute dependency in rough sets?

Answer:

In rough set theory, the notion of dependency is defined very simply. Let us take two (disjoint) sets of attributes, set P and set Q, and inquire what degree of dependency obtains between them. Each attribute set induces an (indiscernibility) equivalence class structure, the equivalence classes induced by P given by $[x]_P$, and the equivalence classes induced by Q given by $[x]_Q$.

Let $[x]_Q = \{Q_1, Q_2, Q_3, \dots, Q_N\}$, where Q_i is a given equivalence class from the equivalence-class structure induced by attribute set Q. Then, the *dependency* of attribute set Q on attribute set P, $\gamma_P(Q)$, is given by

$$\gamma_P(Q) = \frac{\sum_{i=1}^{N} |\underline{P}Q_i|}{|\mathbb{U}|} \leq 1$$

Q.1 e. Describe Ant Colony Algorithms.

Answer:

The **ant colony optimization** algorithm **(ACO)** is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs.This algorithm is a member of the **ant colony algorithms** family, in swarm intelligence methods, and it constitutes some metaheuristic optimizations. Initially proposed by Marco Dorigo in 1992 in his PhD thesis, the first algorithm was aiming to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems have emerged, drawing on various aspects of the behavior of ants.

Some of the variations of ACO are:

- **Elitist ant system**
- **Max-Min ant system (MMAS)**
- **Ant Colony System**
- **Rank-based ant system (ASrank)**
- **Continuous orthogonal ant colony (COAC)**

Q.1 f. Explain Random and Tournament Selection methods of Genetic Algorithm.

Answer:

Sol.) Random Selection
This technique randomly selects a parent from the population. In terms of disruption of genetic codes, random selection is a little more disruptive, on average, than roulette wheel selection.

Tournament Selection
An ideal selection strategy should be such that it is able to adjust its selective pressure and population diversity so as to fine-tune GA search performance. Unlike, the Roulette wheel selection, the tournament selection strategy provides selective pressure by holding a tournament competition among Nu individuals.

The best individual from the tournament is the one with the highest fitness, which is the winner of Nu. Tournament competitions and the winner are then inserted into the mating pool. The tournament competition is repeated until the mating pool for generating new offspring is filled. The mating pool comprising of the tournament winner has higher average population fitness. The fitness difference provides the selection pressure, which drives GA to improve the fitness of the succeeding genes.
This method is more efficient and leads to an optimal solution.

Q.1 g. Explain Particle Swarm Optimization (PSO).

Answer:

In computer science, particle swarm optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. PSO optimizes a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position and is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions.

PSO is a metaheuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc.

Q.2 a. Consider we have three fuzzy sets, given by

$$\underset{\sim}{A} = \left\{ \frac{1}{3} + \frac{0.8}{7} \right\}, \quad \underset{\sim}{B} = \left\{ \frac{0.6}{4} + \frac{1.0}{6} \right\}, \quad \underset{\sim}{C} = \left\{ \frac{0.8}{2} + \frac{1}{4} + \frac{0.4}{8} \right\}.$$

Make suitable decisions based on fuzzy ordering.

Answer:

Using the truth value of inequality, $\underset{\sim}{A} \geq \underset{\sim}{B}$, as follows:

$$\begin{aligned} T(\underset{\sim}{A} \geq \underset{\sim}{B}) &= \max_{x_1 \geq x_2} \left\{ \min(\mu_{\underset{\sim}{A}}(x_1), \mu_{\underset{\sim}{B}}(x_2)) \right\} \\ &= \max\{\min(0.8, 0.6), \min(0.8, 1.0)\} \\ &= \max\{0.6, 0.8\} \\ &= 0.8. \end{aligned}$$

Similarly,

$$T(\underset{\sim}{A} \geq \underset{\sim}{C}) = 0.8, \quad T(\underset{\sim}{B} \geq \underset{\sim}{A}) = 1.0, \quad T(\underset{\sim}{B} \geq \underset{\sim}{C}) = 1.0, \quad T(\underset{\sim}{C} \geq \underset{\sim}{A}) = 1.0,$$
$$T(\underset{\sim}{C} \geq \underset{\sim}{B}) = 0.6.$$

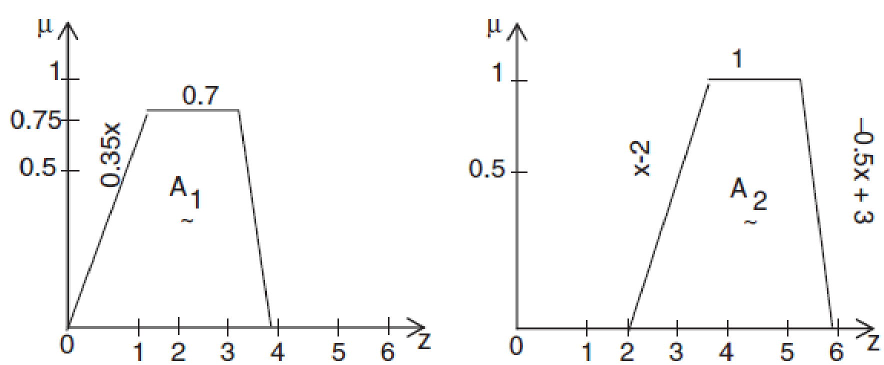
Then,

$$T(\underset{\sim}{A} \geq \underset{\sim}{B}, \underset{\sim}{C}) = 0.8,$$
$$T(\underset{\sim}{B} \geq \underset{\sim}{A}, \underset{\sim}{C}) = 1.0,$$
$$T(\underset{\sim}{C} \geq \underset{\sim}{A}, \underset{\sim}{B}) = 0.6.$$

From this calculation, the overall ordering of the three fuzzy sets would be $\underset{\sim}{B}$ first $\underset{\sim}{A}$ second, and $\underset{\sim}{C}$ third.

Thus the fuzzy ordering is performed

Q.2 b. For the given membership function as shown in Figure below. Determines the defuzzified output value by any Four methods.



Answer:

Centroid method

A11(0, 0), (2, 0.7)

The straight line may be:

$$
\begin{aligned}
& y - 0 = \frac{0.7}{2}(x - 0), \\
& y = 0.35x. \\
& A_{12} : y = 0.7. \\
& A_{13} : \text{not needed.} \\
& A_{21} : (2, 0)(3, 1), \\
& y - 0 = \frac{1 - 0}{3 - 2}(x - 2), \\
& y = x - 2. \\
& A_{22} : y = 1. \\
& A_{23} : (4, 1)(6, 0), \\
& y - 1 = \frac{0 - 1}{6 - 4}(x - 4), \\
& y = \frac{-1}{2}(x - 4) + 1 = -0.5x + 3. \\
& \text{Solving } A_{12} \text{ and } A_{21}, \\
& Y = 0.7 \quad y = x - 2, \\
& x - 2 = 0.7, \\
& x = 2.7, \\
& y = 0.7.
\end{aligned}
$$

$$
\begin{aligned}
\text{Numerator} &= \int_0^2 0.35\ z^2\ \mathrm{d}z + \int_2^{2.7} 0.7K \mathrm{d}z + \int_{2.7}^3 \left(z^2 - 2z\right) \mathrm{d}z \\
&\quad + \int_3^4 z\ \mathrm{d}z + \int_4^6 \left(-0.5z^2 + 3z\right) \mathrm{d}z \\
&= 10.98. \\
\text{Denominator} &= \int_0^2 0.35\ z^2\ \mathrm{d}z + \int_2^{2.7} 0.7K\ \mathrm{d}z + \int_{2.7}^3 \left(z^2 - 2\right) \mathrm{d}z \\
&\quad + \int_3^4 \mathrm{d}z + \int_4^6 \left(-0.5z^2 + 3z\right) \mathrm{d}z \\
&= 3.445.
\end{aligned}
$$

$$z^* \quad = \quad \frac{\text{Numerator}}{\text{Demoninator}} = \frac{10.98}{3.445} = 3.187.$$

(b) Weighted average method

$$z^* = \frac{2 \times 0.7 + 4 \times 1}{1 + 0.7} = 3.176.$$

(c) Mean–max method

$$z^* = \frac{2.5 + 3.5}{2} = 3.$$

(d) Center of saws method

$$\begin{aligned} z^* \quad &= \quad \frac{\int_0^6 \left(\frac{1}{2} \times 0.7 \times (3+2) \times 2 + \frac{1}{2} \times 1 \times (2+4) \times 4\right)}{\int_0^6 \left(\frac{1}{2} \times 0.7 \times (3+2) + \frac{1}{2} \times 1 \times (2+4) \times 4\right)} \\ &= \quad \frac{\int_0^b (3.5 + 12)\,\mathrm{d}z}{\int_0^b (1.75 + 3)\,\mathrm{d}z} = 2.84. \end{aligned}$$

(e) First of maximum
$z^* = 3.$
(f) Last of maxima
$z^* = 4.$
(g) Center of largest area

$$\begin{aligned} \text{Area of } I \quad &= \quad \frac{1}{2} \times 0.7 \times (2.7 + 0.7) = 1.19, \\ \text{Area of } II \quad &= \quad \frac{1}{2} \times 1 \times (2+3) \times \frac{1}{2} \times 0.7 \times = 2.255. \end{aligned}$$

Area of II is larger, So,

$$\begin{aligned} z^* \quad &= \quad \frac{\int_{2.7}^3 \frac{1}{2} \times 0.3 \times 0.3 \times 2.85\,\mathrm{d}z + \int_3^4 1 \times 1 \times 3.5\,\mathrm{d}z + \int_4^6 \frac{1}{2} \times 2 \times 1\,\mathrm{d}z}{\int_{2.7}^3 \frac{1}{2} \times 0.3 \times 0.3\,\mathrm{d}z + \int_4^3 1 \times 1\,\mathrm{d}z + \int_4^6 \frac{1}{2} \times 2 \times 1\,\mathrm{d}z} \\ &= \quad \frac{\int_{2.7}^3 0.12825\,\mathrm{d}z + \int_3^4 3.5\,\mathrm{d}z + \int_4^6 5\,\mathrm{d}z}{\int_{2.7}^3 0.045\,\mathrm{d}z + \int_3^4 \mathrm{d}z + \int_4^6 \mathrm{d}z} \\ z^* \quad &= \quad 4.49. \end{aligned}$$

Q.3 a. Describe Classification of Fuzzy Sets.

Answer:
The fuzzy sets can be classified based on the membership functions. They are:
Normal fuzzy set. If the membership function has at least one element in the universe whose value is equal to 1, then that set is called as normal fuzzy set.
Subnormal fuzzy set*.* If the membership functions have the membership values less than 1, then that set is called as subnormal fuzzy set.
These two sets are shown in Fig. 1.
Convex fuzzy set*.* If the membership function has membership values those are monotonically increasing, or, monotonically decreasing, or they are monotonically increasing and decreasing with the increasing values for elements in the universe, those fuzzy set *A* is called convex fuzzy set.
Nonconvex fuzzy set*.* If the membership function has membership values which are not strictly monotonically increasing or monotonically decreasing or both monotonically increasing and decreasing with increasing values for elements in the universe, then this is called as nonconvex fuzzy set.
Figure 2 shows convex and nonconvex fuzzy set.
When intersection is performed on two convex fuzzy sets, the intersected portion is also a convex fuzzy set.
This is shown in Figure 3.
The shaded portions show that the intersected portion is also a convex fuzzy set. The membership functions can have different shapes like triangle, trapezoidal, Gaussian, etc.

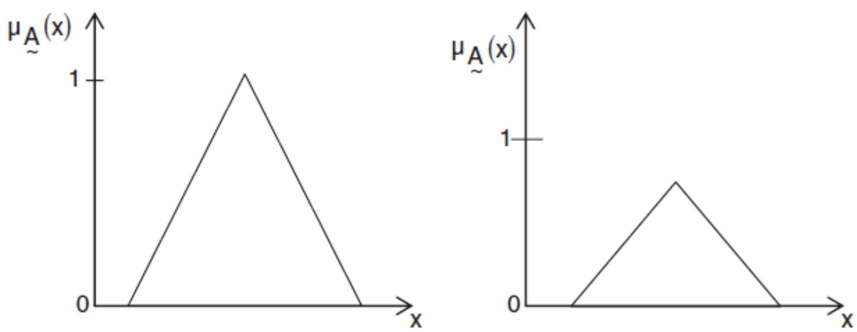


Figure 1. (1) Normal fuzzy set and (2) subnormal fuzzy set

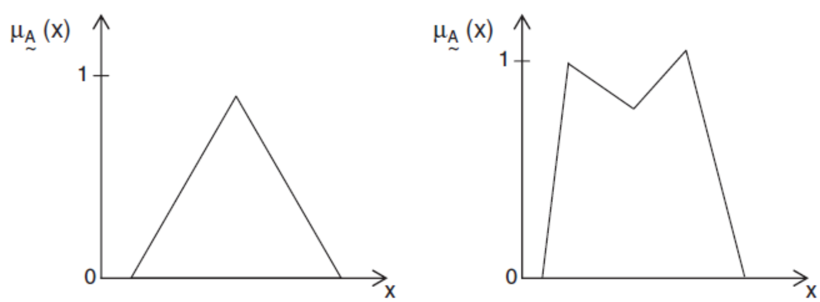


Figure. 2. (a) Convex set and (b) Nonconvex set

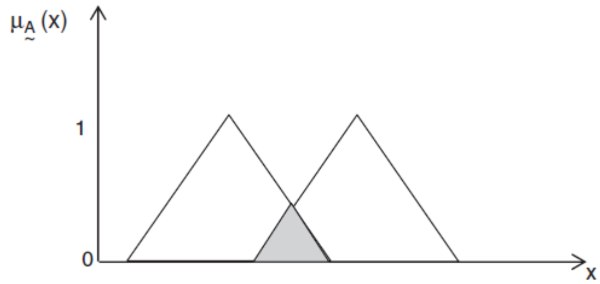


Figure 3 Intersection of two convex sets

Q.3 b. Explain Mutation process of Genetic Algorithm.

Answer:

Sol.) Genetic algorithms have been used for difficult problems (such as NP-hard problems), for machine learning and also for evolving simple programs. They have been also used for some art, for evolving pictures and music. A few applications of GA are as follows:

- Nonlinear dynamical systems–predicting, data analysis
- Robot trajectory planning
- Evolving LISP programs (genetic programming)
- Strategy planning
- Finding shape of protein molecules
- TSP and sequence scheduling
- Functions for creating images
- Control–gas pipeline, pole balancing, missile evasion, pursuit
- Design–semiconductor layout, aircraft design, keyboard configuration, communication networks
- Scheduling–manufacturing, facility scheduling, resource allocation
- Machine Learning–Designing neural networks, both architecture and weights, improving classification algorithms, classifier systems
- Signal Processing–filter design
- Combinatorial Optimization–set covering, traveling salesman (TSP), Sequence scheduling, routing, bin packing, graph coloring.

Q.3 c. Discuss Crossover techniques of Genetic Algorithm

Answer:

Techniques are.

1 One-point crossover
2 Two-point crossover
3 "Cut and splice"
4 Uniform Crossover and Half Uniform Crossover
5 Three parent crossover

Also need to describe each briefly.

Q.4 a. Define Radial Basis Function Networks. Explain the architecture and learning methods of RBFN?

Answer: Page Number 238 of Neuro Fuzzy and Soft Computing

Q.4 b. Explain the procedure of Roulette Wheel Selection method.

Answer:

Sol.) Roulette selection is one of the traditional GA selection techniques. The commonly used reproduction operator is the proportionate reproductive operator where a string is selected from the mating pool with a probability proportional to the fitness. The principle of roulette selection is a linear search through a roulette wheel with the slots in the wheel weighted in proportion to the individual's fitness values. A target value is set, which is a random proportion of the sum of the fit nesses in the population.

The population is stepped through until the target value is reached. This is only a moderately strong selection technique, since fit individuals are not guaranteed to be selected for, but somewhat have a greater chance. A fit individual will contribute more to the target value, but if it does not exceed it, the next chromosome in line has a chance, and it may be weak. It is essential that the population not be sorted by fitness, since this would dramatically bias the selection.

The above described Roulette process can also be explained as follows: The expected value of an individual is that fitness divided by the actual fitness of the population.

Each individual is assigned a slice of the roulette wheel, the size of the slice being proportional to the individual's fitness. The wheel is spun N times, where N is the number of individuals in the population. On each spin, the individual under the wheel's marker is selected to be in the pool of parents for the next generation.

This method is implemented as follows:

1. Sum the total expected value of the individuals in the population. Let it be T.
2. Repeat N times:

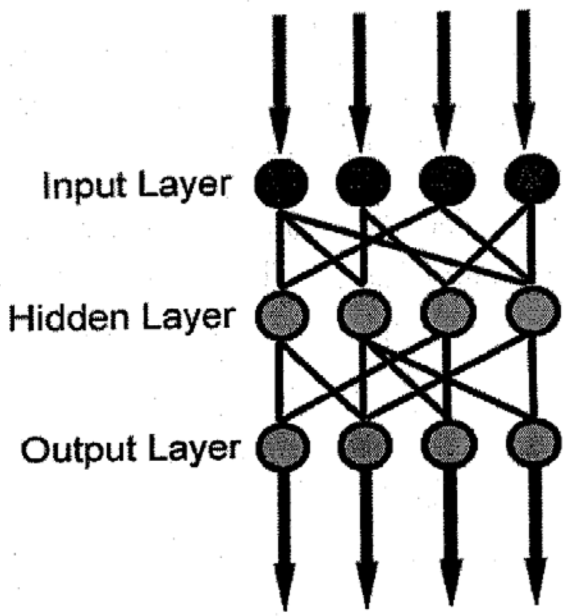
i. Choose a random integer 'r' between o and T.

ii. Loop through the individuals in the population, summing the expected values, until the sum is greater than or equal to 'r'. The individual whose expected value puts the sum over this limit is the one selected.

Roulette wheel selection is easier to implement but is noisy. The rate of evolution depends on the variance of fitness's in the population.

Q.5 a. Compare feed-forward and back-forward neural network.
Answer:

Sol.) A **feed forward neural network** is an artificial neural network where connections between the units do *not* form a directed cycle. This is different from recurrent neural networks.The feed forward neural network was the first and arguably simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network.

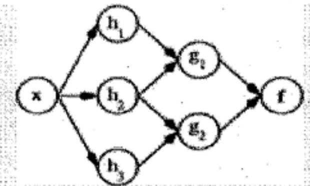


Feedback networks can have signals travelling in both directions by introducing loops in the network. Feedback networks are very powerful and can get extremely complicated. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found. Feedback architectures are also referred to as interactive or recurrent, although the latter term is often used to denote feedback connections in single-layer organisations.

Q.5 b. What is the weight adjustment with sigmoid activation function?

Answer:
Weight adjustment with sigmoid activation function

weighted sum, where $f(x)=K\left(\sum_i w_i g_i(x)\right)$, where K (commonly referred to as the activation function[1]) is some predefined function, such as the hyperbolic tangent. It will be convenient for the following to refer to a collection of functions g_i as simply a vector $g=(g_1,g_2,\ldots,g_n)$.



ANN dependency graph

This figure depicts such a decomposition of f, with dependencies between variables indicated by arrows. These can be interpreted in two ways.

The first view is the functional view: the input x is transformed into a 3-dimensional

- The weight of a connection is adjusted by an amount proportional to the product of an error signal δ, on the unit k receiving the input and the output of the unit j sending this signal along the connection: $\Delta_p w_{jk} = \gamma \delta_k^p y_j^p$.
- If the unit is an output unit, the error signal is given by $\delta_o^p = (d_o^p - y_o^p)\,\mathcal{F}'(s_o^p)$.
  Take as the activation function F the 'sigmoid' function as defined $y^p = \mathcal{F}(s^p) = \dfrac{1}{1+e^{-s^p}}$. In this case the derivative is equal to

$$\begin{aligned}\mathcal{F}'(s^p) &= \frac{\partial}{\partial s^p}\frac{1}{1+e^{-s^p}}\\ &= \frac{1}{(1+e^{-s^p})^2}(-e^{-s^p})\\ &= \frac{1}{(1+e^{-s^p})}\frac{e^{-s^p}}{(1+e^{-s^p})}\\ &= y^p(1-y^p).\end{aligned}$$

such that the error signal for an output unit can be written as: $\delta_o^p = (d_o^p - y_o^p)\,y_o^p(1-y_o^p)$.

- The error signal for a hidden unit is determined recursively in terms of error signals of the units to which it directly connects and the weights of those connections. For the sigmoid activation function:

$$\delta_h^p = \mathcal{F}'(s_h^p)\sum_{o=1}^{N_o}\delta_o^p w_{ho} = y_h^p(1-y_h^p)\sum_{o=1}^{N_o}\delta_o^p w_{ho}.$$

Q.6 a. Write classification of Hybrid function.

Answer:

HYBRID SYSTEMS

ybrid systems are those for which more than one technology is employed to solve the problem. ybrid systems have been classified as (Refer Gray and Kilgour, 1997) 1. Sequential Hybrids, Auxiliary Hybrids, and 3. Embedded Hybrids.

1 Sequential Hybrid Systems

s the name indicates, *sequential hybrid systems* make use of technologies in a pipeline-like ıshion. Thus, one technology's output becomes another's input and so on. Figure 10.1 illustrates ıe schema for a sequential hybrid.

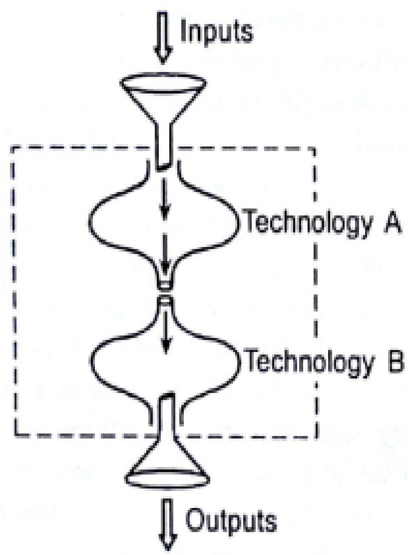


Fig. 10.1 A sequential hybrid system.

This is one of the weakest forms of hybridization since an integrated combination of the technologies is not present.

An example is a GA preprocessor which obtains the optimal parameters for different instances of a problem and hands over the 'preprocessed' data set to an NN for further processing.

.2 Auxiliary Hybrid Systems

In this, one technology calls the other as a "subroutine" to process or manipulate information needed by it. The second technology processes the information provided by the first and hands it over for further use. Figure 10.2 illustrates an auxiliary hybrid system. This type of hybridization though better than sequential hybrids, is considered to be of intermediary level only.

An example is a neuro-genetic system in which an NN employs a GA to optimize its structural parameters, i.e. parameters which defines its architecture.

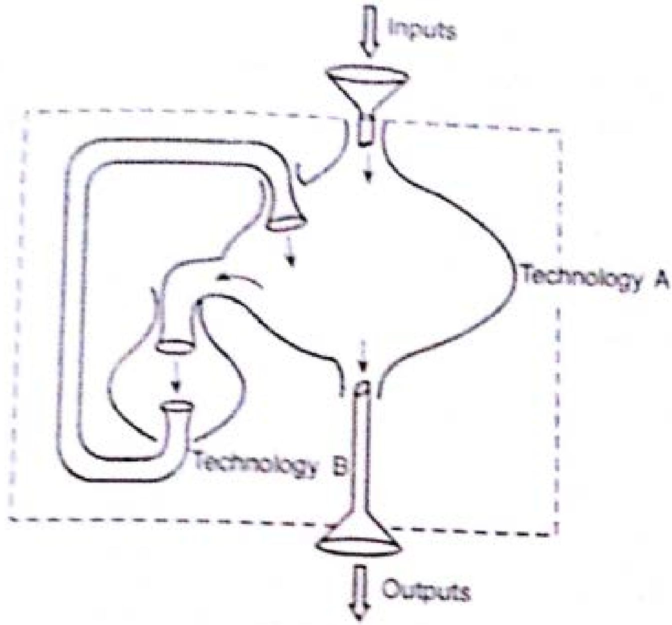
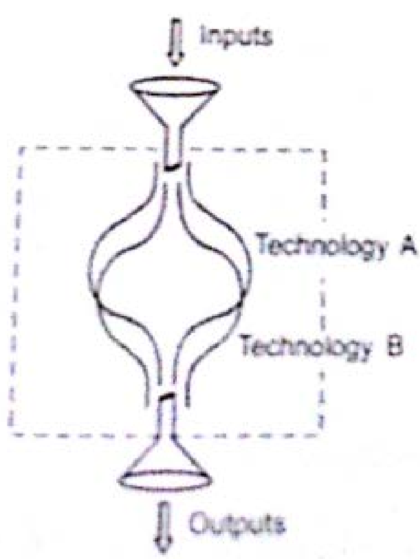


Fig. 10.2 An auxiliary hybrid system.

.3 Embedded Hybrid Systems

In *Embedded hybrid systems*, the technologies participating are integrated in such a manner that they appear intertwined. The fusion is so complete that it would appear that no technology can be used without the others for solving the problem. Figure 10.3 illustrates the schema for an embedded hybrid system. Here, the hybridization is absolute.



Q.6 b. What is Edge Selection and Pheromone update in the ant colony optimization algorithm.

Answer:

An ant is a simple computational agent in the ant colony optimization algorithm. It iteratively constructs a solution for the problem at hand. The intermediate solutions are referred to as solution states. At each iteration of the algorithm, each ant moves from a state x to state y, corresponding to a more complete intermediate solution. Thus, each ant k computes a set $A_k(x)$ of feasible expansions to its current state in each iteration, and moves to one of these in probability. For ant k, the probability p_{xy}^k of moving from state x to state y depends on the combination of two values, viz., the *attractiveness* η_{xy} of the move, as computed by some heuristic indicating the *a priori* desirability of that move and the *trail level* τ_{xy} of the move, indicating how proficient it has been in the past to make that particular move.

The *trail level* represents a posteriori indication of the desirability of that move. Trails are updated usually when all ants have completed their solution, increasing or decreasing the level of trails corresponding to moves that were part of "good" or "bad" solutions, respectively.

In general, the kth ant moves from state x to state y with probability

$$p_{xy}^k = \frac{(\tau_{xy}^{\alpha})(\eta_{xy}^{\beta})}{\sum_{y \in \mathrm{allowed}_y} (\tau_{xy}^{\alpha})(\eta_{xy}^{\beta})}$$

where

τ_{xy} is the amount of pheromone deposited for transition from state x to y, 0 ≤ α is a parameter to control the influence of τ_{xy}, η_{xy} is the desirability of state transition xy (*a priori* knowledge, typically $1/d_{xy}$, where d is the distance) and β ≥ 1 is a parameter to control the influence of η_{xy}. τ_{xy} and η_{xy} represent the attractiveness and trail level for the other possible state transitions.

Pheromone update

When all the ants have completed a solution, the trails are updated by

$$\tau_{xy} \leftarrow (1-\rho)\tau_{xy} + \sum_k \Delta\tau_{xy}^k$$

where τ_{xy} is the amount of pheromone deposited for a state transition xy, ρ is the *pheromone evaporation coefficient* and $\Delta\tau_{xy}^k$ is the amount of pheromone deposited by kth ant, typically given for a TSP problem (with moves corresponding to arcs of the graph) by

$$\Delta\tau_{xy}^k = \begin{cases} Q/L_k & \text{if ant } k \text{ uses curve } xy \text{ in its tour} \\ 0 & \text{otherwise} \end{cases}$$

where L_k is the cost of the kth ant's tour (typically length) and Q is a constant

Q.7 a. Define Rough set and write it's applications.
Answer:
Definition of a *rough set*

Let $X \subseteq \mathbb{U}$ be a target set that we wish to represent using attribute subset P; that is, we are told that an arbitrary set of objects X comprises a single class, and we wish to express this class (i.e., this subset) using the equivalence classes induced by attribute subset P. In general, X cannot be expressed exactly, because the set may include and exclude objects which are indistinguishable on the basis of attributes P.

For example, consider the target set $X = \{O_1, O_2, O_3, O_4\}$, and let attribute subset $P = \{P_1, P_2, P_3, P_4, P_5\}$, the full available set of features. It will be noted that the set X cannot be expressed exactly, because in $[x]_P$, objects $\{O_3, O_7, O_{10}\}$ are indiscernible. Thus, there is no way to represent any set X which *includes* O_3 but *excludes* objects O_7 and O_{10}.

However, the target set X can be *approximated* using only the information contained within P by constructing the P-lower and P-upper approximations of X:

$$\underline{P}X = \{x \mid [x]_P \subseteq X\}$$

$$\overline{P}X = \{x \mid [x]_P \cap X \neq \emptyset\}$$

Applications

Rough set methods can be applied as a component of hybrid solutions in machine learning and data mining. They have been found to be particularly useful for rule induction and feature selection (semantics-preserving dimensionality reduction). Rough set-based data analysis methods have been successfully applied in bioinformatics, economics and finance, medicine, multimedia, web and text mining, signal and image processing, software engineering, robotics, and engineering (e.g. power systems and control engineering).

Q.7 b. Describe Neuro-Fuzzy Hybrid System and Neuro-Genetic Hybrid and System.
Answer:

NEURAL NETWORKS, FUZZY LOGIC, AND GENETIC ALGORITHMS HYBRIDS

In this book, we confine ourselves to hybridization of the three technologies, namely neural networks, fuzzy logic, and genetic algorithms. Neural networks, fuzzy logic, and genetic algorithms are three distinct methodologies each with its own advantages and disadvantages. It is therefore appropriate that a hybridization of the technologies is attempted to overcome the weaknesses of one with the strengths of the other.

1 Neuro-Fuzzy Hybrids

This is one of the most researched forms of hybrid systems and has resulted in a stupendous quantity of publications and research results.

Neural networks and fuzzy logic represent two distinct methodologies to deal with uncertainty. Each of them has its own merits and demerits. Neural networks can model complex nonlinear relationships and are appropriately suited for classification phenomenon into predetermined classes. On the other hand, the precision of outputs is quite often limited and does not admit zero error but only minimization of least squares errors. Besides, the training time required for an NN can be substantially large. Also, the training data has to be chosen carefully to cover the entire range over which the different variables are expected to change.

Fuzzy logic systems address the imprecision of inputs and outputs directly by defining them using fuzzy sets and allow for a greater flexibility in formulating system descriptions at the appropriate level of detail.

Neural networks and fuzzy logic though different technologies, can be used to accomplish the specification of mathematical relationships among numerous variables in a complex dynamic process, perform mappings with some degree of imprecision, in different ways, and can be used to control nonlinear systems to an extent not possible with conventional linear control systems.

Neuro-Fuzzy systems which are an integration of NN and FL have demonstrated the potential to extend the capabilities of systems beyond either of these technologies when applied individually (Haykin, 1994; Kartalapoulos, 1996).

There are two ways of looking at this hybridization. One is to endow NNs with fuzzy capabilities, thereby increasing the network's expressiveness and flexibility to adapt to uncertain environments. The second aspect is to apply neuronal learning capabilities to fuzzy systems to make the fuzzy systems more adaptive to changing environments. This approach is also known, in the literature, as *NN driven fuzzy reasoning* (Takagi and Hayashi, 1991).

2 Neuro-Genetic Hybrids

Neural networks can learn various tasks from training examples, classify phenomena, and model nonlinear relationships. However, the primary features that are of concern in the design of the network are problem specific. Despite the availability of some guidelines, it would be helpful to have a computational procedure in this aspect, especially for the optimum design of an NN. Genetic algorithms have offered themselves as potential candidates for the optimization of parameters of NN (Harp et al., 1989, 1990). The integration of GAs with NNs has turned out to be useful (Schaffer et al., 1992). Genetically evolved nets have reported comparable results against their conventional counterparts (Kitano, 1990; Whitley and Hanson, 1989).

While gradient descent learning algorithms have reported difficulties in learning the topology of the networks whose weights they optimize, GA based algorithms have provided encouraging results especially with regard to face recognition, animation control, and other problems.

Genetic algorithms encode the parameters of NNs as a string of the properties of the network, that is, chromosomes. A large population of chromosomes representing the many possible parameter sets for the given NN is generated. Combined GA-NN technology also known as 'GANN' have the ability to locate the neighbourhood of an optimal solution quicker than other conventional search strategies. But once in the neighbourhood of the optimal solution GANN strategies tend to converge slower than the conventional ones. Drawbacks of GANN algorithms are:

The large amount of memory required to handle and manipulate chromosomes for a given network, and

The question whether this problem scales as the size of the networks becomes large.

Parallel versions of the GA computational paradigm have also been applied on NN (Maniezzo, 1994).

3 Fuzzy-Genetic Hybrids

Fuzzy systems have been integrated with GAs. Kosko (1992) has shown that fuzzy systems like NNs (feedforward) are universal approximators in the fact that they exhibit the capability to approximate general nonlinear functions to any desired degree of accuracy. The adjustment of system parameters that is called for in the process, so that the system output matches the training data, has been tackled using GAs. Several parameters which a fuzzy system is involved with namely input/output variables and the membership functions that define the fuzzy systems, have been optimized using GAs. Nomura et al. (1994) proposed a genetic approach to the problem of fuzzy system adaptation.

Q.7 c. Write applications of Ant colony optimization algorithms.

Answer:
Ant colony optimization algorithms have been applied to many combinatorial optimization problems, ranging from quadratic assignment to protein folding or routing vehicles and a lot of derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations. It has also been used to produce near-optimal solutions to the travelling salesman problem. Others are

- Scheduling problem
- Vehicle routing problem
- Assignment problem
- Set Problem
- Data mining
- Discounted cash flows in project scheduling
- Distributed Information Retrieval
- Grid Workflow Scheduling Problem
- Image processing

Text Books

1. S.R. Jang, C.T. Sun and E. Mizutani, Neuro-Fuzzy and Soft Computing, Pearson Education 2004

2. Andries P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, John Wiley and Sons, 2007