International Journal of Research in Advent Technology, Vol.2, No.5, May 2014
E-ISSN: 2321-9637

Application of Neural Network in SOCN for Decision-Making

Priyanka Podutwar, Ms. V. M. Deshmukh
M.E. CSE, Prof., M.E. CSE
PRMIT&R Badnera
Podutwar13@gmail.com, msvmdeshmukh@rediffmail.com

Abstract-An Artificial Neural Network (ANN) is gives an information about the biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is used in a specific application, such as pattern recognition or data classification, through a learning process. Advantages of neural network include Self-organization occurs in a variety of physical, chemical, biological, social and cognitive systems. Common examples are crystallization, the emergence of convection patterns in a liquid heated from below, chemical oscillators, swarming in groups of animals, and the way neural networks learn to recognize complex patterns. Self-Organization, Real Time Operation, Fault Tolerance via Redundant Information Coding, Adaptive learning. Neural networks also used in other areas of research like neurology and psychology. Neural network has number of applications such as, medicine and business, the AI, fuzzy logic.

A self-organizing computing network based on concepts of fuzzy conditions, beliefs, probabilities, and neural networks is proposed for decision-making in intelligent systems which are required to handle data sets with a diversity of data types. A sense-function with a sense-range and fuzzy edges is defined as a transfer function for connections from the input layer, the hidden layer and output layer in the network. Hidden cells adjust the parameters of the sense-functions. Input layer provides computing cells, which are used for designing networks for data converters, can deal with both symbolic data and numeric data. Hidden computing cells in the network can be explained via fuzzy rules in fuzzy neural networks. The values in the output layer can be explained as a belief distribution over a decision space. The final decision is made by using the winner-take-all rule. The approach was applied to a series of the benchmark data sets with a diversity of data types and comparative results obtained. Based on these results, the suitability of a range of data types for processing by different intelligent techniques was analyzed, and the results show that the proposed approach is better than other approaches for decision-making in information systems with mixed data types.

Index Terms- Intelligent system; decision support; machine learning; fuzzy condition; fuzzy sets.

1. INTRODUCTION

In order to automatically make decisions, an intelligent system may incorporate and apply many existing techniques such as decision trees, Bayesian belief networks, evidence theory, rough set theory, fuzzy set theory, kNN (k-nearest-neighborhood) classifier, neural networks, and support vector machines. We call the technique "intelligent" if it includes three important aspects: knowledge, decision-making, and learning, i.e., when an intelligent system interacts with its environment, it can automatically make correct decisions by means of its knowledge, and it can learn from instances to gain knowledge. Each intelligent technique has its own knowledge representation, decision-making, and learning mechanisms.

Knowledge-based or Artificial Intelligence techniques are used increasingly as alternatives to more classical techniques to model environmental systems. We review some of them and their environmental applicability, with examples and a reference list. The techniques covered are case-based reasoning, rule-based systems, artificial neural networks, fuzzy models, genetic algorithms, cellular automata, multi-agent systems, swarm intelligence, reinforcement learning and hybrid systems.

If knowledge is represented by a feed forward neural network, the network can be trained by the back propagation algorithm with a training set, and outputs of the trained network can be applied to make decisions. Because of differing representations of knowledge and differing mechanisms of learning and decision-making, a particular intelligent technique is usually good at dealing with a specific data type. For example, neural networks are good at dealing with numeric data while decision trees are good at dealing with symbolic data.

Two classes of data, symbolic data and numeric data, are often used in an intelligent system. Symbolic data values are discrete, and relationships among the values are not explicitly indicated. It is difficult to measure a distance between a pair of symbolic data values for example, suppose that the data attribute food names contains values:"pizza", "burgers", "noodles", "manchurians". It is difficult to say which .Furthermore, value sequences or neighborhoods are not explicitly indicated among symbolic values. For value is the nearest neighbor of value "foody". The values cannot be sorted in a descending sequence or an ascending sequence in the same way as numbers. Symbolic data are called S-type data in this paper. Numeric data can be classified into two types: continuous values and codes. Continuous values, which are called C-type data in this paper, can be addressed by mathematical functions. C-type values are not independent. There is a sequence among the values, and each pair of values can be measured with a distance. Values which are a short distance away from value A can be called the neighbors of value A. C-type data intrinsically contain knowledge about value sequences and numeric labels, and they are called N-type data in this paper. They are usually represented by integers, for example, room numbers, ID numbers, item numbers, or other label numbers that are encoded by humans. They are somewhat similar to C-type data in that there is a sequence among N-type values, and the distance between the values can be measured. However, the distance between two values is not always significant. For example, it is not guaranteed that a student with ID = 001 and a student with ID = 002 are neighbors. The N-type data can therefore be regarded as either S-type or C-type. The choice depends on whether a distance between values is significant for decision-making.

Each intelligent technique [2][3] has its own merits and shortcomings. For example computational intelligent techniques, such as dynamic fuzzy neural networks, kNN, neural networks, and support vector machines, are good at dealing with C-type data. However, they cannot be directly applied to S-type data or to a data set with missing values. Symbolic AI techniques are good at dealing with S-type data and data with missing values. However, if they are applied to C-type data, the C-type data have to be discretized , and the results are affected by the discretization. In the real world, an intelligent system often encounters mixed data types, incomplete information (missing values), and imprecise information (fuzzy conditions). In the UCI Machine Learning Repository, it can be seen that there are many real-world data sets with missing values and mixed data types .It is a very important challenge to enable a machine leaning or data mining approaches to deal with mixed data types. In a discretizer based on class-dependence is applied to transform C-type data to S-type data and enable inductive learning and decision tree learning systems to deal with C-type data. A symbolic AI technique plus a good discretizer is the traditional machine-learning approach to handle an information system with mixed data types. However, information about distance and neighborhood in C-type data is ignored, if symbolic AI techniques treat the discretized values as symbolic values. In order to apply information about distance and neighborhoods, Multimodal Data Fusion was investigated, and difficulties in finding a measure of similarity between objects with mixed data type attributes were detailed. These difficulties are shared by machine learning in data mining approaches. To date, an effective solution has not been recorded in literature. Therefore, a novel Self-Organizing Computing Network is proposed here to deal with data sets with mixed data types. The concepts of support degree and belief in symbolic AI and concepts in fuzzy neural networks are applied to this computing network to avoid the difficulties indicated. As this network contains many concepts that do not belong to conventional neural networks, the term computing network is applied for clarity. In Chapter 3, the structure of the computing network proposed to deal with data sets with mixed data types is described. In Chapter 4, a set of algorithms is given for self-organization and decision-making. In Chapter 5, the benchmark data sets from UCI Machine Learning Repository are used to test the algorithms and an analysis of the experimental results is given. Chapter 6 presents our conclusion.

2. LITERATURE

2.1 *History*

The concept of self-organization can be traced back to at least two sources: Western philosophy influenced heavily by Greek thinking; and eastern philosophy, centered around the process thinking of Bhuddism. From both of these sources ideas have been generated which resound with the modern way of thinking about self-organization although the word itself had never been used. On wondering about the origin of the world Greek and Roman atomists from Democritus to Epicurus argued that world order arose from chance collisions of particles. First, the cosmos (from Greek kosmos = the ordered) did not exist but chaos instead (from Greek chaos = the disordered). In modern times chaos theory has taken up this topic again, with deep connections to ideas about self organization and the origin of order in the universe.

The first use of the term:

Work on General Systems Theory (von Bertalanffy) and Cybernetics (Wiener) paved the way for the idea of self-organization. The concept of a self-organizing system was introduced by Ashby in 1947. In the

1950s a self-organizing system was considered to be a system which changes its basic structure as a function of its experience and environment. The term was used by Farley and Clark in 1954 to describe learning and adaptation mechanisms. Ashby, in 1960, redefined a self-organizing system to include the environment with the system proper. Von Foerster argued, also in 1960, that a dynamical system which is self-organizing possesses some stable structures (eigenvalues, attractor states) which he later termed eigenbehavior.

Further developments:
This notion was further developed by Haken in 1977 who termed the global cooperation of elements of a dynamical system in the attractor state which emerges from the interaction between these elements and the environment self organization. Both Haken and Kauman (1993) argued for a deep connection between self-organization and selection. Kauman in particular emphasized the role of constraints on the direction of evolution caused by self-organization. Already in the 1970s, however, ideas branched out into different directions. One branch of the development of the idea deepened the relation to studies of learning and adaptation (Conrad, Kohonen), another branch studied processes of self-organization in systems far from equilibrium (Prigogine, Haken). Chaostheory (Thom, Mandelbrot) was the line of inquiry into nonlinear systems in mathematics, whereas autopoiesis and self-maintenance where at center stage in biology (Eigen, Rosen) neurophysiology and cognitive science (von Foerster, Maturana, Varela). In recent years, self-organizing systems have assumed center stage in the natural sciences, and the humanities. Engineering is beginning to see the usability of the concept in connection with the approach of nano-scale applications.

2.2 Need
In order to automatically make decisions, an intelligent system may incorporate and apply many existing techniques such as decision trees, Bayesian belief networks, evidence theory, rough set theory, fuzzy set theory, kNN (k-nearest-neighborhood) classifier, neural networks, and support vector machines. We call the technique “intelligent” if it includes three important aspects: knowledge, decision-making, and learning. If knowledge is represented by a feed forward neural network, the network can be trained by the back propagation algorithm with a training set, and outputs of the trained network can be applied to make decisions. Because of differing representations of knowledge and differing mechanisms of learning and decision-making, a particular intelligent technique is usually good at dealing with a specific data type.

2.3 Purpose

In Science:
Self-organization as a concept has assumed center stage in Science. With the advent of nonlinear systems and studies on complex systems in non-equilibrium situations, the explanatory power of self-organization permeated every branch of scientific enquiry. From structure formation at the level of super-galactic clusters, even starting from the development of the entire universe, down to microscopic particles and their interaction patterns, self-organizing phenomena have been postulated, theorized, observed and conformed. In particular the origin and evolution of life have been studied under the aspect of self-organization. Within Biology, the developmental process of organisms as well as their metabolisms, growth and learning have been identified as self-organizing processes.

In the Humanities:
In the humanities, the idea of self-organization has taken roots, although the paradigm is far from being fully recognized yet. Since the 1990s the origin and development of languages has been an object of study under the premise of self-organization. Laszo has put forward the notion of self-organization in human history. In social science, due to the obvious interaction between actors, the concept of self-organization has been earlier considered. Even in psychology, self-organizing principles begin to appear. Economy and Management Science have taken notice of the concept, and more and more recommendations are published for the purpose of propagating this point of view. Finally, Philosophy has embraced the concept of self-organization and connected it to earlier thoughts on the development of the scientific process and Epistemology. Whitehead put forward his process philosophy, and Smuts, already in the 1920s, promoted the notion of holism which has strong connections to self-organization.

In Engineering:
Engineering is beginning to grasp the ubiquity of self-organization in Nature. Specifically in the area of nanotechnology the concept is used extensively for the purpose of self-assembly of molecular entities. At nanoscales, it is very difficult to directly specify the structuring behavior of entities. As a result, self-organizing properties of matter are used to the advantage of the structural outcome. In the area of adaptation, there exists a long tradition of making use of self organization principles. The self-organizing feature map, introduced by Kohohen, has been a key step forward in the domain of unsupervised learning of artificial neural networks.

3. COMPUTING NETWORK MODEL
The standard advantaged claimed for neural networks are multiple computing units, parallel processing and adaptive learning mechanisms. Concepts of fuzzy set theory have been applied to neural networks to create fuzzy neural networks [4]. The concepts in symbolic

AI techniques are applied to design a self-organizing computing network so that the network is capable of dealing with mixed data types.

The self-organizing computing network is defined in this paper is constructed by means of three classes of computing cells: input cells, hidden cells, and output cells. In general, computing cells can have different computing mechanisms; for a self-organizing computing network, input cells are defined as a set of encoders that transform different data values to the internal representation of the network. An input cell corresponds to an attribute in an instance information system. Therefore, the number of input cells is equal to the number of attributes in the instance information system. Each input cell is connected to hidden cells by means of connection transfer functions. Hidden cells are self-organized by means of known instances in the instance information system. The number of hidden computing cells is determined by the self-organizing algorithm. Hidden computing cells are connected to output cells according to the decision values of known instances in the instance information system. The number of output cells is equal to the number of decision values of the decision attribute in the instance information system.

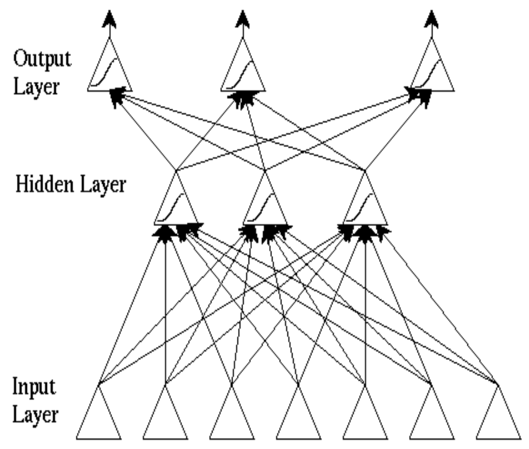

Fig 3.1: Computing network

The architecture shown in Fig. 3.2 is a self-organizing computing network for an instance information system with p attributes and n decision values.

Let I = $\{ i_1,........,i_q,i_s,.......i_p \}$ be a set of input cells, $I_C = \{i_1,......, i_q\}$ for C-type (or N-type) inputs and $I_S = \{i_S ,......, i_p\}$ for S=type inputs. Let $H = \{h_1, h_2,....h_m\}$ be a set of hidden cells, $K = \{k_1, k_2,.......k_n\}$ be a set of output cells. The connections from input cells to hidden cells are represented by a transfer functions matrix $T_{I\times H}$.

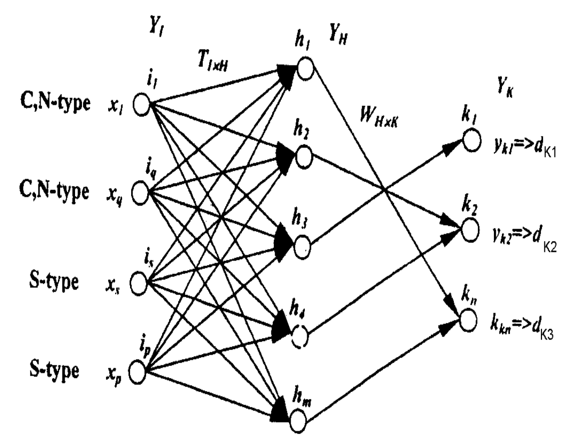

Fig 3.2: Architecture of a computing network

$$T_{I\times H} = \begin{Bmatrix} T_{i1,h1} & T_{i1,h2} & .... & T_{i1,hm} \\ & .... & & \\ T_{ip,h1} & T_{ip,h2} & .... & T_{ip,hm} \end{Bmatrix} \qquad (3.1)$$

Connections between the hidden layer and the output layer are weighted connections. The weights are represented by the following matrix.

$$W_{H\times K} = \begin{Bmatrix} w_{h1,k1} & w_{h1,k2} & w_{h1,kn} \\ & .... & \\ w_{hm.k1} & w_{hm,k2} & w_{hm.kn} \end{Bmatrix} \qquad (3.2)$$

The weights are real numbers with a range of [0, 1]. A weight in the weight matrix can be regarded as a decision support degree for a hidden cell. For example Weight $w_{h,k} = 0$ means that hidden cell h does not support the decision corresponding to output cell k, if it is equal to 1 then it supports the decision corresponding to output cell k.

3.1: Input Cells-Input Data Encoders

In a computing network as shown in Fig.3.1, an input cell is defined as an input data encoder that transforms input data with type diversity to the internal representation. Let $X_I = \{x_{i1} ,...., x_{iq} , x_{is} ,...., x_{ip}\}$ represent input values of the input cells and $Y_I = \{y_{i1} ,...., y_{iq} , y_{is} ,...., y_{ip}\}$ represent output values of the input cells. In order to transform all C-type and N-type inputs to the same value range, a computing mechanism for C-type and N-type input cells is represented by the following expressions:

$$y_i = \left(x_i - x_{i(\min)} \right) s_i \quad for \ i \in I_C \qquad (3.1.1)$$

$$s_i = \left(s_i(\max) - s_i(\min)i \right) s_I \quad for \ i \in I_C \qquad (3.1.2)$$

Where [0, S_W] is the value range and S_W can be an arbitrary integer or real number . If X_1 is an S-type value, an encoding scheme is required. If the S-type data can be arranged in a significant sequence, the S-type value is converted to the order number or rank in the sequence. For example, suppose temperature has values: "hot", "warm", and "cold". The sequence "cold->warm->hot" is a significant sequence. And, it represents the temperature varying from low to high. S-type data for the temperature can be encoded with N-type data, 1 for "cold", 2 for "warm", and 3 for "hot". Equations (3.1.1) and (3.1.2) are applied to transform these values to the internal representation of the computing network. If S-type data cannot be arranged in a significant sequence, S-type data are transformed to integer codes arbitrarily. For example, values for food names can be encoded with 1 for "pizza", 2 for "burger", 3 for "noodles", 4 for "muncurians". They could also be encoded in other codes.

$$y_i = code \quad for\ i \in I_s \ , i.e., S\_type\ inputs \qquad (3.1.3)$$

In order to get a redundant integer for these codes, Sw is set to a typical value A real number within [0, Sw] is applied to the internal representation of C-type and N-type input values.

3.2 Hidden Cells-Sense-functions

Hidden cells receive data from the outputs of input cells through connections between them. Each connection transforms data from input cells to hidden cells through the transfer function matrix as shown in (3.1). Let $y_H = \{y_{h1,....,} y_{H,......,} y_{hm}\}$ be a set of the outputs of hidden cells, $y_h(Y_{I_C})$ be the combination of all C-type inputs and $y_h(Y_{I_s})$ N-type inputs and be the combination of all S-type inputs. An output of hidden cell is defined as

$$y_h(Y_I) = y_h(Y_{I_C})y_h(Y_{I_S}) \qquad (3.2.1)$$

As C-type data have continuous values, the transfer function is a sense-function with a specific response sense-range. For simplicity, a general sense-function is denoted as follows:

$$\varphi\ (y\ ,\ L\ ,U\ ,\delta\ )$$

where ,y is a variable with continuous values, constant L is the lower edge, constant U is the upper edge, and constant δ represents the sharpness (i.e., the "vagueness degree") of the edges. The range [L;U] is the sense-range of the function.

The left panel in Fig.3.2.1 shows two curves with the same constant δ = 10 and with different sense-ranges [10, 20] and [30, 40].

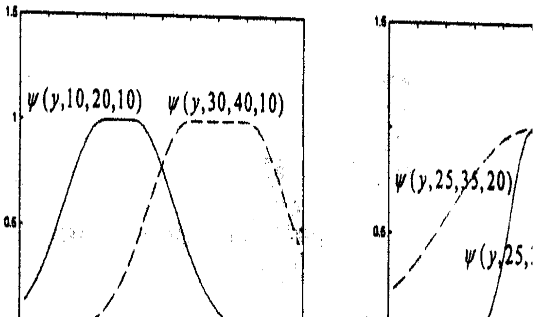

Fig.3.2.1 Different sense-range and edge sharpness for sense-function

The right panel in Fig.3.2.1 shows two curves with the same sense-ranges [25, 35] and with different constant δ = 20 and δ = 5. The larger the constant is, the lower the sharpness (greater the vagueness degree) of the edges is. We can apply this sense-function the combination of C-type and N-type inputs.

Fig.3.2.2 shows that a hidden computing cell receives C-type inputs from afferent cells. Each connection from C-type inputs in the input layer to the hidden layer is indicated by three constants $(L_{i,h}, U_{i,h}, \delta_{i,h})$

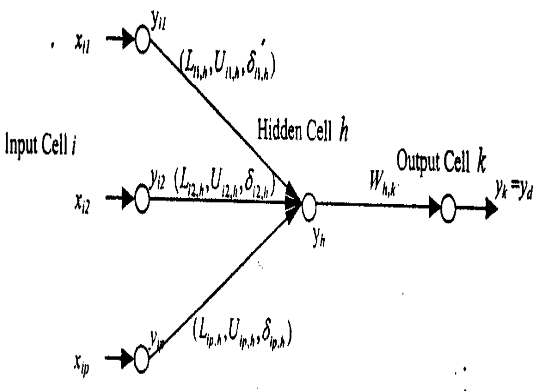
y_{i1}
x_{i1}
$(L_{i1,h},U_{i1,h},\delta_{i1,h})$
Hidden Cell h
Input Cell i
Output Cell k
y_{i2} $(L_{i2,h},U_{i2,h},\delta_{i2,h})$
x_{i2}
$W_{h,k}$
$y_k = y_d$
y_h
y_{ip}
$(L_{ip,h},U_{ip,h},\delta_{ip,h})$
x_{ip}

Fig.3.2.2 A hidden computing cell receives C-type inputs from different cells

3.3 Competitive Output Layer for Decision Making

An output value of a hidden cell is a product of C-type condition match degree and S-type support degree. A weight from a hidden cell to output cell can be regarded as a support degree. Therefore, the product $W_{h,k}\ y_h$ can be regarded as the degree of belief that the input data support the decision d associated with the output cell k. Each output cell may be connected to multiple hidden cells by $W_{H \times K}$ .corresponding to the hidden cell that supports decision d. The maximal belief is defined by maximal belief,

$$y_{d\,(\max)} = \max_{h \in H} (w_h, dY_h) = y_k \qquad (3.3.1)$$

where, y_h is an output of hidden cells. The output is regarded as the computing mechanism of output cells.

4. ALGORITHMS FOR SELF-ORGANIZING COMPUTING NETWORK

In order to illustrate how to generate a computing network by means of self-organizing algorithms, the example instance information system shown in Table 4.1 is applied.

Table 4.1: Instance Information System for Fitting Contact Lenses

u	x_{i1}	x_{i2}	x_{i3}	x_{i4}	d
1	1	1	1	1	3
2 *	1	1	1	2	2
3	1	1	2	1	3
4	1	1	2	2	1
5	1	2	1	1	3

The data set is from the UCI Machine Learning Repository. There are 24 instances in the instance information system with four attributes and one decision attribute with three values. Column x_{i1} is an attribute for the age of patients: "1" for young, "2" for prepresbyopic , and "3" for presbyopic. Column x_{i2} is an attribute for the spectacles prescription: "1" for myope and "2" for hypermetrope. Column x_{i3} is an attribute for astigmatism: "1" for no , "2" for yes. Column x_{i4} is an attribute for the year production rate: "1" for reduced and "2" for normal. Column d is the decision attribute:

- "1"-the patient should be fitted with hard contact lenses.
- "2"-the patient should be fitted with soft contact lenses.
- "3"-the patient should be fitted with contact lenses.

In order to compare this network with other machine learning approaches, a 10-fold cross-validation standard is applied. The 10 folds of instances for this case are derived by means of the following equations:

$$FOLD_K = \{INSTANCE_{Nou} : k = \mathrm{mod}(u/10)\} \qquad (4.1)$$

According to the 10-fold cross-validation standard, one fold is regarded as an unseen test set and the other nine folds are regarded as a training set. u is an instance. An average accuracy of the 10-fold cross validation can be obtained by using 10 folds. For example, the instances with "*" in Table 4.1 belong to Fold 2, which is regarded as an unseen test set and is represented by

$$TS = FOLD_2. \qquad (4.2)$$

The instances in TS cannot be applied for training. The corresponding training set is represented

$$TR = \{INSTANCE_{Nou} : \mathrm{mod}(u/10) \neq 2\}. \qquad (4.3)$$

4.1 Initial Network

The input layer is automatically generated according to the number of attributes and the sampled values of attributes in an instance information system. Hidden cells can be self-organized by means of the instances in a training set. Two steps are required to transform from an instance into a hidden cell. First, the condition values of the instance are transformed to transfer functions of connections between input cells and the hidden cell. Second, the connection weights between the hidden cell and output cells are set up according to the decision value of the instance. Actually, the number of hidden cells is much less than the number of instances in the system. The goal of this self-organizing algorithm is to maximize the fuzzy condition subsuperspace adapted to training sets by means of adjusting the sense-ranges of the sense-functions. Initially, an instance can be randomly chosen to generate the first hidden cell. In order to clearly illustrate this self-organizing algorithm, suppose that all the attributes in Table 4.1 are regarded as C-type.

4.2 Adjusting Sense-Function Parameters

The larger the sense-ranges for transfer functions in the condition, the more situations can be covered by the hidden cell. Based on instances in a training set, the largest sense-ranges for a hidden cell can be found by adjusting the lower edge and the upper edge. The algorithm for finding maximal upper edges and minimal lower edges can be represented by equations as follows:

$$L_{\min} = \min\{L : y_h(u) < \theta \quad for \quad u \in ND\}, \qquad (4.2.1)$$

$$U_{\max} = \max\{U : y_h(u) < \theta \quad for \quad u \in ND\} \qquad (4.2.2)$$

Because $y_h(u)$ represents the value of the condition match degree for the cell that supports decision d, the value should be lower than θ .ND represents instances that do not support d.

4.3 Self-organizing Hidden Layer

Definition: There is an instance u that satisfies

$$y_h(u) \geq (1 - \varepsilon),$$

i.e., the condition vector of instance u is matched with the hidden cell Ch with a tolerance constant. This indicates that the instance has been covered by the hidden cell Ch. The tolerance constant is a small number. It can be applied to control the coverage of the hidden cell. The value is set to 0.05 in our experiments.

Based on this definition, the new sense-ranges cover the conditions in instances No.1,3,5,7,9,11,13,15,17,19,21 and 23.

All the instances in training set TR from table 4.1 can be covered by subsuperspaces of seven hidden cells. During the self-organization of hidden cells, the weight matrix is obtained as follows:

$$W_{H \times O} = \left\{ \begin{matrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{matrix} \right\}$$

This approach for generating hidden cells is called a self-organizing algorithm.

4.4 Decision Based on the Network

The formal algorithm for decision-making is as follows:

Algorithm for decision-making*:*

1. Get input vector $(x1, x2, ...xp)$ from an instance u.
2. Convert input vector to the internal representation $Y_1(u)$ by using the input cell computing mechanism
3. Apply transfer functions and computing mechanism of hidden layer cells to the condition vector $Y_1(u)$ to obtain the hidden layer outputs $Y_H = (y1, y2, ... ym)$
4. Apply the hidden layer outputs Y_H and connections $W_{H \times K}$ obtain a belief distribution over the decision space.
5. Apply the winner-take-all rule to get a decision.

$$d_M = \arg \max_{d \in Vd} (y_{d\,(\max)})$$

The time complexity for this algorithm is O(|H|) , where |H| is the number of hidden cells. There is one problem in using winner-take-all rule; a situation may arise in which there are multiple decisions with equal maximal belief values. A reasonable decision can be made by means of the remaining transfer functions.

5. Conclusion

The Self-Organizing Computing Network represents a combination of symbolic AI techniques and computational intelligent approaches. In this approach, the input layer is designed as a set of data converters. The Bayes classifier is applied to input cells for converting from probabilities of symbolic data to decision support degrees. Based on the concepts of receptive fields and the synapse response function in biological neurons, a sense-function with a sense-range is defined for each connection from input cells to hidden cells. If the inputs are symbolic data, the sense-function is replaced by a support degree distribution. Each hidden cell is defined as a multiplier that combines sense functions or support degree distributions from the afferent cells. Each hidden cell responds to a fuzzy subsuperspace within the input superspace. This is similar to a biological neuron having receptive field. Just as neural receptive fields adapts to stimuli, the fuzzy subsuperspace of a hidden cell adapts to the training set according to the self-organizing algorithms. Based on the definition for the computing cell and its connections, each output value can be explained using symbolic AI concepts such as belief, support degree, match degree, or probability. Using this network, instance information system with mixed data types can be handled directly. Note that preparation of input data is included in the self-organizing algorithm. Therefore, this approach has not directly given results for data reduction and aggregation, and structural and semantic heterogeneity. It may be possible to extend the cells in the hidden layer to retrieve these.

REFERENCES

1) IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING. VOL.18, NO. 7, JULY 2006.
2) J.R. Quinlan, “Improved Use of Continuous Attributes in C4.5,” *J. Artificial Intelligent Research,* vol. 4, pp. 77-90, Mar. 1996.
3) N. Kasabov, *Envolving Connectionist Systems-Methods and Applications, Brain Study and intelligent Machines.* London : Springer-Verlag, 2003.
4) W.L. Tung and C. Quek, “GenSoFNN : A Generic Self-Organizing Fuzzy Neural Network,” *IEEE Trans. Neural Networks,* Vol. 13, no. 5, pp. 1075-1086, Sept. 2002.
5) IEEE Transaction on Neural Networks22(2):199-210(2011)
6) A Survey on Transfer Learning. IEEE Trans .Knowl. Data Eng. 22(10): 1345-1359 (2010)
7) "An on-line algorithm for creating self-organizing fuzzy neural networks" [Neural Networks 17(10)(2004)1477-1493]
8) A Self-Organizing Computing Network for Decision-Making in Data Sets with a Diversity of Data Types. IEEE Trans. Knowl. Data Eng. 18(7): 941-953 (2006)
9) IEEE Trans. Neural Netw. Learning Syst. 23(5): 689-702 (2012)
10) An on-line algorithm for creating self-organizing fuzzy neural networks. Neural Networks 17(10): 1477-1493 (2004)
11) A Distribution-Index-Based Discretizer for Decision-Making with Symbolic AI Approaches. IEEE Trans. Knowl. Data Eng. 19(1): 17-28 (2007)
12) An approach for on-line extraction of fuzzy rules using a self-organizing fuzzy neural network. Fuzzy Sets and Systems 150(2): 211-243 (2005)
13) Q.X. Wu, D.A. Bell, and T.M. McGinnity, “Multi-Knowledge for Decision Making,” *Int'l J. Knowledge and Information Systems*, vol. 7, no. 2, pp. 246-266, Feb. 2005.
14) J.Y. Ching, A.K. C. Wong, and K.C. C. Chan, “Class-Dependent Discretization for Inductive Learning from Continuous and Mixed-Mode Data,” *IEEE Trans. Pattern Analysis and*

Machine Intelligence, vol. 17, no. 7, pp.641-651,July1995.

15) S. Coppock and L. Mazlack, “Soft Multi-Modal Data Fusion,” *Proc. Conf. Fuzzy Systems (FUZZ '03),* vol. 1, pp. 25-28, May 2003.

16) T. Takagi and M. Sugeno, “Fuzzy Identification of Systems and Its Applications to Modeling and Control,” *IEEE Trans. Systems, Man, and Cybernetics,* vol.15,no.1,pp.116-132,1985.

17) J.S. R. Jang and C.T. Sun, “Functional Equivalence between Radial Basis Function Networks and Fuzzy Inference Systems,” *IEEE Trans. Neural Networks,* vol.4,no.1,pp156-159,1993.

18) W. Gerstner and W.M. Kistler, *Spiking Neuron Models. Single Neurons, Populations,Plasticity.* CambridgeUniv.Press, 2002.

19) M.J. Chacron, B. Doiron, L. Maler, A. Longtin, and J. Bastian, “Non-Classical Receptive Field Mediates Switch in a Sensory Neuron's Frequency Tuning,” *Nature,* vol.423,pp.77-81,May2003.

20) U.T. Eden, L.M. Frank, R. Barbieri, V. Solo, and E.N. Brown, “Dynamic Analysis of Neural Encoding by Point Process Adaptive Filtering,”*Neural Computation,* vol.16,no.5,pp.971-998,2004.

21) A. Hodgkin and A. Huxley, “A Quantitative Description of Membrane Current and Its Application to Conduction and in Nerve,” *J. Physiology,* vol. 117, pp. 500-544, 1952.