www.icgst.com

Designing of an Efficient Classifier using Hierarchical Reinforcement Learning

Moumita Saha[1], *Jaya Sil*[1], *Nandita Sengupta*[2]
Computer Science And Technology, Bengal Engineering And Science University, Shibpur, West Bengal, India[1]
Information Technology Program, University College of Bahrain,P.O. Box 55040, Manama, Kingdom of Bahrain[2]

Abstract

Large dimensional real life dataset often consists of vague and redundant information, creating difficulty in building an efficient classifier. In the early part of this work, fuzzy-rough set and genetic algorithm (GA) were applied on continuous domain to select important features sufficient to classify the dataset, called reducts. The aim of the paper is to generate fuzzy rule set with optimum variation in the range of linguistic labels representing antecedent of the rules. Hierarchical Q-Learning method consisting of two levels, are applied to achieve the goal. In the first level of learning, optimized variation is achieved with respect to each reduct and in the second level, learning is applied to optimize the variation with respect to each attribute of the reduct. The performance of the classifiers are evaluated before and after learning, demonstrating improvement in classification accuracy by imparting training.

Keywords: *Q Learning, Hierarchical Q Learning.*

1 Introduction

Data analysis is an important area of research, needs attention of computer scientists in order to extract knowledge by efficient handling of the large, inexact dataset. In continuous domain, selection of minimal feature set and their values in optimized intervals play significant role in in improving classification accuracy. Authors in [1] proposed fuzzy Q-learning optimization technique for traffic load balancing in GSM Edge Radio Access Network. In this paper, simulation results proved that application of fuzzy Q-learning optimization in load balancing reduces the call block considerably. In [2], fuzzy Q-learning controller is used to adapt SHO parameters to varying network situations such as traffic fluctuation. Combination of fuzzy logic and reinforcement learning simplifies dynamic optimization of fuzzy logic rules which yields better online SHO parameterization of each base station in the network. Fei Liu et. al. proposed [3] combination of reinforcement mutation and genetic algorithm, RMGA for solving Traveling Salesman Problem (TSP). They have established with experimental results that RMGA produces optimum timing of each tour of TSP for any size of problem. Mechanism of combining of genetic algorithm and reinforcement learning for optimization problem has been addressed in [4]. More knowledge through learning of the environment is provided to genetic algorithm, applied on TSP.

Q learning is a widely used reinforcement learning technique [5]-[12], suitable for online applications. The learning algorithm executes best possible action in a particular state to reach to the goal state, assigned by the agent(s). However, the flat structure reinforcement learning (Q-Learning) suffers from computational complexity with increase of number of state variables in the problem domain. Hierarchical reinforcement learning is designed to deal with such problem. The whole problem is subdivided into hierarchical level so that curse of dimensionality can be avoided. Order of execution of such subtasks depend on the requirement of the system. Reward of each subtask in any of the hierarchical level contributes to the total reward of learning. Termination condition of each subtask form the goal of learning. There are many advantages in hierarchical reinforcement learning. First, as number of variables is reduced in each hierarchical level, learning can be faster as few trials are needed. Secondly, learning for any subtask, in any level of hierarchy can be reused in any other problem. Hierarchical reinforcement learning is based on Semi Markov Decision Process ($SMDP$) which is an extension of $Markov\ Decision\ Process$.

In our earlier work [13], fuzzy-rough set concept was applied to avoid discretization of data, resulting no information loss. GA was applied on the continuous dataset to reduce dimensionality of data and selecting important features, called reducts [14]-[21]. To classify the data, fuzzy rule set is derived where range of linguistic levels representating the antedecents of rules are initialized by analyzing the dataset. Then, different variation to the range of the linguistic levels is enforced by evaluating standard deviation of the dataset using statistical methods. Hierarchical Q-Learning algorithm has been applied in the paper to obtain optimized variation in the range of linguistic levels of the rules both in the reduct and in the individual attribute levels. The performance of the classifiers are evaluated before and after learning, demonstrating improvement in classification accuracy by imparting training.

The paper is organized within five sections, where Section 2 explains preparation of input data, Section 3 describes proposed hierarchical Q-Learning algorithm, Section 4 discusses the experimental results and finally Section 5 presents conclusion of the work.

2 Data Preparation

Data preparation compromises of two major steps. First, initial assignment of range of values for the linguistic levels and secondly evaluation of the variation for applying as input to the hierarchical Q -Learning.

2.1 Initial assignment of Range

- The training dataset is rearranged according to their class-labels.
- For each attribute and each class-labels the minimum and maximum attribute values are noted.
- A linguistic level is assigned for each range between the minimum and maximum attribute values of every attribute.

2.2 Preparation of Variation

The following steps are adopted in order to evaluate the variation for selecting the range of fuzzy variables present in the rule set and used as dimension in Q-Matrix.

- First the standard deviation (sd_total) of the total dataset and the same for each of the conditional attributes (sd_attr_i) are calculated.
- Now as there are n number of conditional attributes, mean of the standard deviation (sd_attr) of each attribute is calculated.
- The deviation (dev) between sd_total and sd_attr is evaluated.
- The sd_attr is considered as the starting variation, and the other variations are obtained using equation (1).

$$var_{i+1} = sd_attr + (i * dev) \qquad (1)$$

where i=1,2,...n.

3 Optimization of Range

The main aim of this work is to evaluate an optimal range for each linguistic levels of the attributes, in order to build a rule-base classifier that maximises the accuracy of classification.

3.1 Hierarchical Q Learning Algorithm

In the paper, the proposed Hierarchical Q learning algorithm has been applied to Wine dataset to improve the accuracy of classification, where the number of class labels are three. Two levels in Hierarchical Q Learning are denoted by $level\_1$ and $level\_2$ and executed to attain the goal.

3.1.1 Developing Reward Matrix

In the proposed Q-learning algorithm, the Reward matrix is developed in two phases- (i) Initial Reward Matrix and (ii) Final Reward Matrix. In $level\_1$, each variation is mapped as row (i.e. the state) and each reduct as column (i.e. the action). Classification rules are derived for individual reducts and accuracy corresponding to each reduct is calculated using the rule based classifier. Accuracy is thresholded to frame the Initial Reward matrix with three discrete values [-1, 0, 1], representing different kind of actions, as defined in equation (2). Next the result of $level\_1$ is utilised to frame *initial reward matrix* of $level\_2$ of the Hierarchical Q Learning. In $level\_2$, the optimised variation to each reduct (as calculated from the previous level) is considered as states and the attributes as actions. The initial reward matrix is evaluated in same way as in $level\_1$.

$$\begin{aligned} r_{ij} &= \text{-1 if Accuracy}(Red_i) < 65\% \text{ or if attribute } j \notin Red_i \\ &= \text{0 if } 65\% < \text{Accuracy}(Red_i) < 75\% \qquad (2) \\ &= \text{1 if } 75\% < \text{Accuracy}(Red_i) < 100\% \end{aligned}$$

The structure of Initial Reward matrix (R) is shown below.

$$R = \begin{bmatrix} 1 & -1 & -1 & ...... & 1 \\ -1 & 0 & 0 & ...... & 0 \\ ... & ... & ... & .... & ... \\ 1 & 1 & 1 & ...... & 1 \end{bmatrix}$$

Algorithm Initial Reward Matrix (R)

Repeat
Step 1: The initial Reward Matrix R(n X m) consisting of elements initial- rij = -1 /0 /1 .
m represents the variation while n represents reduct in $level\_1$ and m represents the variation along with reduct while n represents attributes in $level\_2$

Step 2: Generate reduct set, say Red_i where i=1..q

(number of reduct). Find red_b where
Accuracy (red_b)=maximum(Accuracy (red_1),
Accuracy (red_2)
.....Accuracy(red_q)).

Step 3: The values of the matrix is assigned using equation (2)

Initial Reward Matrix for $level\_1$, R

$$
\begin{array}{c|cccccc}
 & Red_1 & Red_2 & Red_3 & Red_4 & Red_5 & Red_6 \\
\hline
St0-Var0 & -1 & -1 & -1 & -1 & -1 & -1 \\
St1-Var1 & -1 & -1 & 0 & 0 & -1 & -1 \\
St2-Var2 & -1 & 0 & 0 & 0 & -1 & -1 \\
St3-Var3 & -1 & -1 & 1 & 1 & -1 & -1 \\
St4-Var4 & -1 & 1 & 1 & 1 & -1 & -1
\end{array}
$$

From R, the Final Reward matrix (RF) is obtained by eliminating the columns, having -1 in all rows. Dimension of RF matrix is now determined, which is less compared to R.

Algorithm Final Reward Matrix (RF)

Input : Initial Reward Matrix R(n × m)
Output: Final Reward Matrix RF(n × p)
where p <= m.

Step 1: For each m, ascertain wheather all r_{ij} are -1.

```
a = 0 ;
    Begin for ( j= 1 to m )
        Counter = 0 ;
        Begin for ( i=1 to n)
            if ( r_ij == -1 )
                Counter++ ;
        End for

        if ( counter == number-of-cuts )
            Set Deleted-column[a++] = j ;
End for
```

Step 2: Remove the selected c ∈ Deleted-column[ ] having all r_{ij} as -1.

```
p = 0;
    Begin for ( j= 1 to m )
        flag = 0 ;
        Begin for ( k= 1 to a )
            Check if ( i== Deleted-column[k] )
                Set flag = 1 ;
        End for

        if ( flag== 0 )
            Begin for ( i= 1 to n )
                r_ip = r_ij ;
            End for
            p + =1;
    End for
```

Step 3: Replace the element -1 with -1.

```
Begin for ( i = 1 to n )
    Begin for ( j = 1 to p )
        checkif ( r_ip == -1 )
            r_ip = -1 ;
        End for
    End for.
```

Final Reward Matrix for $level-1$, RF

$$
\begin{array}{c|ccc}
 & Red_2 & Red_3 & Red_4 \\
\hline
St0-Var0 & -1 & -1 & -1 \\
St1-Var1 & -1 & 0 & 0 \\
St2-Var2 & 0 & 0 & 0 \\
St3-Var3 & -1 & 1 & 1 \\
St4-Var4 & 1 & 1 & 1
\end{array}
$$

3.1.2 Modified Q-Learning Algorithm

From the Final Reward matrix, Q-matrix has been developed. Start state of Q-matrix corresponds to a particular range of linguistic label and the goal state is defined as the state at which maximum accuracy is achieved. At goal state, optimal range of values to linguistic labels are obtained. All rows excluding the last row (goal state) of the Q matrix comprising of all zeroes. The last row corresponds to the goal state consists all ones representing maximum classification accuracy. Learning procedure consisting of several episodes and continues till all the elements in the modified Q matrix become greater than '0' representing accuracy.

BEGIN

Input : Final Reward Matrix RF (n × p).

Step 1: Initialize all the elements of the Q-Matrix, QM (n × p) to zero.

Step 2: Assign values to the $n-th$ state (i.e. Goal State) of the QM.

```
For ( j = 1 to p )
QM [ n ] [ j ] = RF [ n ] [ j ] ;
End
```

Step 3: Derive a sparse matrix SM (r × 3) from RF (n × p) which keeps track of i (where i=1,2,.......n), j (where j=1,2,.........p) and assign 0 / 1 in correspondence to r_{ij} of RF.

Step 4: Noting down the i (where i=1,..n) having no action.

```
no-action-size = 0 ;
Begin for ( i = 1 to n )
```

```
Flag2 = 0 ;
  Begin for ( j = 1 to p )
      if( RF [ i ] [ j ] ≥ 0 )
        Flag2 = 1 ;
  End for
      if ( Flag2 == 0 )
        no-action [ no-action-size ++ ] = i ;
End for
```

Step 5: Initialize the Flag[ ] to 0.

Step 6: Starting running the episodes.

```
Begin do while
   Count = 0;
   /*Starting operation from i=0 (i.e. start state) and
   continuing operation until i=n (i.e. Goal State)
   is attained */

   Begin while ( SM [ count ] [ 0 ] ! = ( n-1 ))
       State = SM [ count ] [ 0 ] ;
         Begin if ( ( SM [ count ] [ 2 ] == 0 )and
               ( Flag [state ] == 0 ))
                     action-number
                       = SM [ count ] [ 1 ] ;

   Calculate MAX [ QM ( next-state , all-actions )]

   Update the Q − Matrix
         QM[ state ][ action-number ]
                 =( RF[ state ][ action-number ]
                       + ( ϒ * Max ));

         Update sparse matrix SM ( r × 3 )

         Reinitialise the Flag[ ] array to 0.

         /* Checking wheather all values of QM[ ][ ]
         has been updated */
               Flag-end = 0 ;
         Begin for ( k = 1 to a )
              if( SM [ k ] [ 2 ] == 0 )
                 Flag-end = 1 ;
              End for

         End do while ( Flag-end == 1 ) ;
```

Step 7: Output the Q-Matrix, QM (n × p) .

END

Finally the Q matrix is formed where for each j, the highest q_{ij} value is marked representing i is the optimum variation value for reduct Red_j. Initial Q matrix (QM), Final Q matrix (Q_{final}) and Q matrix (QM) are shown below.

Initial Q matrix for $level\_1$, QM

$$Q = \begin{array}{c} \\ St0-Var0 \\ St1-Var1 \\ St2-Var2 \\ St3-Var3 \\ St4-Var4 \end{array} \begin{array}{c} \begin{array}{ccc} Red_2 & Red_3 & Red_4 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

Final Q matrix for $level\_1$, Q_{final}

$$Q_{final} = \begin{array}{c} \\ St0-Var0 \\ St1-Var1 \\ St2-Var2 \\ St3-Var3 \\ St4-Var4 \end{array} \begin{array}{c} \begin{array}{ccc} Red_2 & Red_3 & Red_4 \end{array} \\ \begin{pmatrix} 1 & 0.8 & 1 \\ 1.64 & 1 & 1.44 \\ 0.8 & 1.8 & 1.64 \\ 1.44 & 1.8 & 1.8 \\ 1 & 1 & 1 \end{pmatrix} \end{array}$$

After evaluating the Q-Matrix, the result i.e. the optimised range of values of linguistic labels is utilized to design the fuzzy rule set. Finally, a rule-base classifier is built and evaluated using the test dataset to measure the performance of the classifier.

4 Experimental Results

In the paper, the proposed Hierarchical Q-Learning is applied on the Wine dataset with 13 attributes. Using our earlier work [13], the reduct set is evaluated, shown in table 1.

Table 1: Reduct Set with Attributes

Reduct	Attributes
Red1	6, 11, 12
Red2	6, 10, 11, 12
Red3	4, 5, 6, 12
Red4	2, 6, 10, 12
Red5	5, 6, 10, 11, 12
Red6	4, 6, 10, 11, 12
Red7	1, 5, 6, 11, 12
Red8	5, 6, 9, 10, 11, 12

Then the variation is claculated using equation (1), shown in table 2.

Table 2: Variation And Its Value

Variation	Values
Var0	0.00000
Var1	0.20610
Var2	0.22830
Var3	0.25041
Var4	0.27252
Var5	0.29463
Var6	0.31674
Var7	0.33885
Var8	0.36096

In $level\_1$ of Hierarchical Q- learning, we have optimised the variation applied to each of the reducts. The initial reward matrix and final Q matrix are shown in table 3, table 4 respectively.

Table 3: Initial Reward Matrix in $level\_1$

Variation	Reducts (Red_i)							
	Red1	Red2	Red3	Red4	Red5	Red6	Red7	Red8
Var0	+1	+1	0	+1	+1	0	+1	0
Var1	+1	+1	+1	+1	+1	+1	0	0
Var2	+1	+1	+1	+1	+1	+1	0	0
Var3	+1	0	+1	-1	0	0	0	0
Var4	+1	0	+1	-1	0	0	0	0
Var5	+1	0	+1	-1	0	0	0	0
Var6	+1	0	+1	-1	0	0	0	0
Var7	+1	0	+1	-1	0	0	0	0
Var8	+1	+1	+1	0	+1	+1	0	0

The final Reward Matrix is same as the Initial Reward Matrix, since there is no column consisting of all -1's in the corresponding rows.

Table 4: Final Q-Matrix in $level\_1$

Variation	Reducts (Red_i)							
	Red1	Red2	Red3	Red4	Red5	Red6	Red7	Red8
Var0	1.0	1.8	1.44	2.952	2.952	2.361	3.3616	2.3616
Var1	1.0	1.8	2.44	2.44	2.952	2.952	1.952	1.952
Var2	1.0	1.8	1.8	2.44	2.44	2.44	1.44	1.44
Var3	1.0	0.8	1.8	0.0	1.44	1.44	1.44	1.44
Var4	1.0	0.8	1.8	0.0	1.7216	1.7216	1.7216	1.7216
Var5	1.0	0.8	2.152	0.0	1.952	1.952	1.952	1.952
Var6	1.0	1.44	2.44	0.0	1.44	1.44	1.44	1.44
Var7	1.8	0.8	1.8	0.0	0.8	0.8	0.8	0.8
Var8	1.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0

Now maximising the goal of each row of Final Q-Matrix, we get the variation, applied to each of the reduct , shown in table 5 which has been utilised to proceed to $level\_2$.

Table 5: Optimal Variation For Reduct

Variation	Reduct
Var0	Red7
Var1	Red5
Var2	Red4
Var3	Red3
Var4	Red3
Var5	Red3
Var6	Red3
Var7	Red1
Var8	Red6

$Level\_2$ optimises the variation, applied to each of the attributes. The variations along with the best reduct (found in $level\_1$) are considered as the states and the attributes on which variations are to be applied as the actions. The evaluation result of *initial reward matrix*, *final reward matrix* and *final Q matrix* are shown in table 6, table 7 and table 8 respectively.

Table 6: Initial Reward Matrix in $level\_2$

Reduct With Variation	Attribute Numbers ($Attr_i$)												
	Attr0	Attr1	Attr2	Attr3	Attr4	Attr5	Attr6	Attr7	Attr8	Attr9	Attr10	Attr11	Attr12
Var0-Red7	-1	+1	-1	-1	-1	+1	+1	-1	-1	-1	-1	+1	+1
Var1-Red5	-1	-1	-1	-1	-1	+1	+1	-1	-1	-1	+1	+1	+1
Var2-Red4	-1	-1	+1	-1	-1	-1	+1	-1	-1	-1	+1	-1	+1
Var3-Red3	-1	-1	-1	-1	+1	+1	+1	-1	-1	-1	-1	-1	+1
Var4-Red3	-1	-1	-1	-1	+1	+1	+1	-1	-1	-1	-1	-1	+1
Var5-Red3	-1	-1	-1	-1	+1	+1	+1	-1	-1	-1	-1	-1	+1
Var6-Red3	-1	-1	-1	-1	+1	+1	+1	-1	-1	-1	-1	-1	+1
Var7-Red1	-1	-1	-1	-1	-1	-1	+1	-1	-1	-1	-1	+1	+1
Var8-Red6	-1	-1	-1	-1	+1	-1	+1	-1	-1	-1	+1	+1	+1

Now, the Final Reward Matrix is formed as shown in table 7, by eliminating the column from table 6 which have all its elements as -1.

Table 7: Final Reward Matrix in $level\_2$

Reduct With Variation	Attribute Numbers ($Attr_i$)							
	Attr1	Attr2	Attr4	Attr5	Attr6	Attr10	Attr11	Attr12
Var0-Red7	+1	-1	-1	+1	+1	-1	+1	+1
Var1-Red5	-1	-1	-1	+1	+1	+1	+1	+1
Var2-Red4	-1	+1	-1	-1	+1	+1	-1	+1
Var3-Red3	-1	-1	+1	+1	+1	-1	-1	+1
Var4-Red3	-1	-1	+1	+1	+1	-1	-1	+1
Var5-Red3	-1	-1	+1	+1	+1	-1	-1	+1
Var6-Red3	-1	-1	+1	+1	+1	-1	-1	+1
Var7-Red1	-1	-1	-1	-1	+1	-1	+1	+1
Var8-Red6	-1	-1	+1	-1	+1	+1	+1	+1

Table 8: Final Q-Matrix For $level\_2$

Reduct With Variation	Attribute Numbers ($Attr_i$)							
	Attr1	Attr2	Attr4	Attr5	Attr6	Attr10	Attr11	Attr12
Var0-Red7	1.0	0.0	0.0	1.8	2.44	0.0	2.952	3.361
Var1-Red5	0.0	0.0	0.0	1.0	1.8	2.44	2.952	3.361
Var2-Red4	0.0	1.0	0.0	0.0	1.8	2.44	0.0	2.952
Var3-Red3	0.0	0.0	1.0	1.8	2.44	0.0	0.0	2.952
Var4-Red3	0.0	0.0	1.0	1.8	2.44	0.0	0.0	3.361
Var5-Red3	0.0	0.0	1.0	1.8	2.95	0.0	0.0	2.952
Var6-Red3	0.0	0.0	1.0	2.44	2.44	0.0	0.0	2.44
Var7-Red1	0.0	0.0	0.0	0.0	1.8	0.0	1.8	1.8
Var8-Red6	-1.0	-1.0	1.0	-1.0	1.0	1.0	1.0	1.0

Now maximising the goal of each column of the Final Q-Matrix, the variation are obtained for each of the attribute in order to improve the accuracy of classification , shown in table 9 .

Table 9: Optimal Variation For Attributes

Attributes	Variation
Attr0	-
Attr1	Var0
Attr2	Var2
Attr3	-
Attr4	Var3/ Var4/ Var6/ Var8
Attr5	Var6
Attr6	Var5
Attr7	-
Attr8	-
Attr9	-
Attr10	Var1
Attr11	Var0
Attr12	Var4

A comparative study of classification accuracy before learning and after learning with no-variation and the optimised range of variation to the linguistic labels of each attributes are evaluated, showing improvement in performance, given in table 10.

Table 10: Comparison Of Accuracy with No-Variation And Optimal-Variation

Reduct	Accuracy (No-Variation)%	Accuracy (Optimal-Variation)%
Red1	81.63	81.21
Red2	78.50	79.50
Red3	70.50	79.50
Red4	78.00	75.00
Red5	77.50	79.50
Red6	74.50	79.00
Red7	76.50	75.50
Red8	74.00	74.00

5 Conclusion

In this paper, concept of Hierarchical Q Learning has been highlighted and a modified Q Learning Algorithm is applied in two levels. The proposed method optimises the range of values of linguistic labels of each attribute. It also focuses on building of an efficient classifier that is suitable for dynamic rule-base, with optimised value range of each attribute. It is also ascertained that accuracy of classification after optimising the linguistic labels either improves or is compatible.

References

[1] P. Munoz, R. Barco, I. de la Bandera, M. Toril, S. Luna-Ramirez, *Optimization of a Fuzzy Logic Controller for Handover-Based Load Balancing*, IEEE 73rdVehicular Technology Conference (VTC Spring), (2011).

[2] R. Nasri, Z. Altman, H. Dubreil, Z. Nouir, *WCDMA Downlink Load Sharing with Dynamic Control of Soft Handover Parameters*, IEEE 63rd Vehicular Technology Conference, VTC 2006-Spring, (2006).

[3] Fei Liu,Guangzhou Zeng, *Study of genetic algorithm with reinforcement learning to solve the TSP*, Elsevier journal of Expert Systems with Applications, Volume 36, Issue 3, Part 2, Pages 6995-7001, April (2009).

[4] F.C. de Lima, J.D. de Melo, A.D.D. Neto, *Using Q-learning Algorithm for Initialization of the GRASP Metaheuristic and Genetic Algorithm*, International Joint Conference onNeural Networks, (2007).

[5] K.-L.A Yau, P. Komisarczuk, *P.D. Teal, Performance analysis of Reinforcement Learning for achieving context-awareness and intelligence in Cognitive Radio networks*, In proceedings of the The 34th annual IEEE conference on Local Computer Networks (LCN), 2009.

[6] Yingzi Wei, Mingyang Zhao, *Composite rules selection using reinforcement learning for dynamic job-shop scheduling* , In proceedings of the 2004 IEEE Conference on Robotics, Automation and Mechatronics.

[7] Chin-Teng Lin, Chong-Ping Jou, *GA-based fuzzy reinforcement learning for control of a magnetic bearing system*, IEEE Transactions on Systems, Man and Cybernetics, Volume 30 Issue 2, pp 276 - 289, 2000.

[8] J. Valasek, J. Doebbler, M.D. Tandale, A.J. Meade, *Improved AdaptiveŰReinforcement Learning Control for Morphing Unmanned Air Vehicles*, IEEE Transactions on Systems, Man and Cybernetics, Volume 38 Issue4, pp. 1014 - 1020 , 2008.

[9] Matthias Rungger, Hao Ding, Olaf Stursberg, *Multiscale Anticipatory Behavior by Hierarchical Reinforcement Learning, Anticipatory Behavior in Adaptive Learning Systems*, Springer-Verlag, Computer Speech and Language, pp. 301-320, 2009.

[10] Andrew G. Barto, Sridhar Mahadevan, *Recent Advances in Hierarchical Reinforcement Learning*, Discrete Event Dynamic Systems , vol 33, pp 41-77, 2003.

[11] Bhaskara Marthi, David Latham, Stuart Russell, Carlos Guestrin, *Concurrent Hierarchical Reinforcement Learning*, In proceedings of the 19th international joint conference on Artificial intelligence, IJCAI'05, 2005.

[12] Thomas G. Dietterich, *Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition*, Journal of Artificial Intelligence Research, volume 13,pp 227-303, 2000.

[13] Moumita Saha, Jaya Sil, *Dimensionality Reduction Using Genetic Algorithm And Fuzzy-Rough Concepts*, 2011 World Congress on Information and Communication Technologies, IEEE, (2011).

[14] Qinghua Hu, Daren Yu, ZongxiaXie, Information-preserving hybrid data reduction based on fuzzy-rough techniques, Pattern Recognition Letters, Vol. 27, pp 414-423, (2006)

[15] Richard Jensen, QiangShen, *Semantics-Preserving Dimensionality Reduction: Rough and Fuzzy-Rough-Based Approaches*, IEEE Transactions On Knowledge And Data Engineering, Vol. 17, No. 1 (2005).

[16] R. Jensen, Q. Shen, *Fuzzy-Rough Attribute Reduction with Application to Web Categorization, Fuzzy Sets and Systems*, vol. 141, no. 3, pp. 469-485, (2004).

[17] Hu Guohua, Shi Yuemei, *An Attribute Reduction Method Based on Fuzzy-Rough Sets Theories*, First International Workshop on Education Technology and Computer Science, (2009).

[18] Richard Jensen, QiangShen, *Rough and Fuzzy Sets for Dimensionality Reduction*, (2001).

[19] Richard Jensen, Andrew Tuson, QiangShen, *Extending propositional satisfiability to determine minimal fuzzy-rough reducts*, FUZZ-IEEE (2010).

[20] Richard Jensen, Chris Cornelis, *Fuzzy-rough instance selection*, FUZZ-IEEE (2010).

[21] Neil MacParthalain, Richard Jensen, *Measures for unsupervised fuzzy-rough feature selection*, Int. J. Hybrid Intell. Syst. 7(4): 249-259 (2010).

Biographies

Moumita Saha has completed her Bachelor of Technology from Meghnad Saha Institute Of Technology, WBUT. She is pursuing Master Of Engineering from Bengal Engineering And Science University(BESUS), Shibpur. She is performing her research work in the field of Soft Computing.

Jaya Sil an alumnus of BESUS(Bengal Engineering and Science University, Shibpur) and JU(Jadavpur University), completed her Ph.D. in Engineering from JU, Kolkata, India. She has been working in Academics for last 25 years and presently working as Professor in the Department of Computer Science and Technology Department in BESUS. She has more than 80 publications in International Conferences and Journals. Dr. Sil worked as Post-Doc Fellow in Nanyang Technological University, Sngapore on 2002-03 and visited Heidelburg University, Germany on 2007. Dr. Sil contributed several Book Chapters on different areas like data mining, image processing. Her areas of research include Image Processing, Soft Computing Techniques, Multiagent Systems and Bio-Informatics.

Nandita Sengupta has done her Bachelor and Master of Engineering from Bengal Engineering and Science University, Shibpur, India. She has 21 years of working experience and last 10 years with academics in University College of Bahrain, Bahrain. Her area of interest is Analysis of Algorithm, Theory of Computation, and Network Computing.