Comparative Review of Fuzzy Rules Generation

Siti Zaiton Mohd Hashim, Dr
University Teknologi Malaysia
Faculty of Computer Science and Information System
81310 UTM Skudai, Johor, Malaysia
+607-5565044
sitizaiton@utm.my

Kurniawan Eka Permana
Trunojoyo University
Departement of Informatic Engineering
Jl. Raya Telang PO. Box 2 Kamal, Indonesia
+62-31-30110
kurniawan@trunojoyo.ac.id

ABSTRACT

Fuzzy rules are usually generated by experts in the area, especially for control problems with only a few inputs. With an increasing number of variables, the number of rules is increasing exponentially, which makes more difficult for experts to define the rule set for good system performance. In solving this, researchers have looked into hybrid the fuzzy based to enhance or reduce complexity. We reviewed two Fuzzy Rules Generations, which is constructed based on Fuzzy Neural Network and Fuzzy Particle Swarm Optimization, respectively. The main section which are used for generation are rule base and membership function. This paper describes each method how to tune the membership function and fuzzy rules as the main phase, step by step.

Keywords
Membership Function, Fuzzy Rules Generation, Fuzzy Neural Network, Fuzzy Particle Swarm Optimization.

1. INTRODUCTION

Fuzzy systems are successfully used in a wide number of applications such as self-focusing cameras, cement-mixing system, vehicle controls (including antilock braking systems) and others. These systems can be considered as knowledge-based systems, incorporating human knowledge into their knowledge base through fuzzy inference system and fuzzy membership functions. The definition of these fuzzy inference system and fuzzy membership functions is generally affected by subjective decisions, having a great influence over the performance of the system. In most existing applications, the fuzzy rules are generated by experts in the area, especially for control problems with only a few inputs. With an increasing number of variables, the number of rules is increasing exponentially, which makes more difficult for experts to define the rule set for good system performance.

Some small fuzzy system conducted employing expert experience, but a lot of application can’t be obtained directly. This has promoted research on how to generate fuzzy rules automatically.

Generally, extracting knowledge based on artificial neural network or fuzzy neural network consists of three main phases. One is to design an adequate network structure that may contain possible configurations of all rules and easy to be trained. Due to fuzzy systems are designed to operate with linguistic variables, the combination of fuzzy systems and neural networks has shown their effectiveness in knowledge acquisition.

Another phase is to use an algorithm for training and pruning the network. The conventional training algorithms are gradient decent method and evolutionary computing method. The goal of network pruning is to make a fat network slim and make the network manageable for knowledge extracting task. The third phase is to acquire knowledge and explain rules after pruning the network.

Another approach formulated the fuzzy system as a space search, where each point represents a rule set and Membership Functions. This makes evolutionary algorithms, such as genetic algorithms (GA), Particle Swarm Optimization (PSO), good candidates for searching these spaces [2-3]. The main idea is based on the food-searching-behavior of birds.

It is observed that they take into consideration the global level of information to determine their direction. Hence, the global and local best positions are computed at each instant of time (iteration), and the output is the new direction of search. Once this direction is detected, it is followed by the cluster of birds searching these spaces. In this paper, we analysis of Fuzzy Rules Generation which is constructs based on Fuzzy Neural Network and FPSO, respectively.

This paper is organized as follows: Section II describes neural network and PSO algorithm. Section III presents the Fuzzy Rules Generation. Section IV shows the conclusion.

2. NEURAL NETWORK AND PSO ALGORITHM

2.1 Neural Network

There are a large number of different types of networks, but they all are characterized by the following components: a set of nodes, and connections between nodes.

Figure 1 show picture of Neural Network. The nodes can be seen as computational units. They receive inputs, and process them to obtain an output. This processing might be very simple (such as summing the inputs), or quite complex (a node might contain another network...). The connections determine the information flow between nodes. They can be unidirectional, when the information flows only in one sense, and bidirectional, when the information flows in either sense.

The interactions of nodes though the connections lead to a global behaviour of the network, which cannot be observed in the elements of the network. This global behaviour is said to be emergent. This means that the abilities of the network supersede the ones of its elements, making networks a very powerful tool.
2.2 Particle Swarm Optimization

The Particle Swarm Optimization Algorithm (PSO) is a population-based optimization method that finds the optimal solution using a population of particles [2, 3].

Every swarm of PSO is a solution in the solution space. PSO is basically developed through simulation of bird flocking in two-dimension space. The PSO definition is presented as follows:

- Each individual particle \( i \) has the following properties:
  - A current position in search space, \( x_{id} \), a current velocity, \( v_{id} \), and a personal best position in search space, \( p_{ib} \).
  - The personal best position, \( p_{ib} \), corresponds to the position in search space where particle \( i \) presents the smallest error as determined by the objective function \( f \), assuming a minimization task.
  - The global best position denoted by \( p_{gd} \) represents the position yielding the lowest error amongst all the \( p_{ib} \).

The velocity and positions of each particle are updated according to their best encountered position and the best position encountered by any particle according to the following equation:

\[
\begin{align*}
  v_{id} &= w \cdot v_{id} + c_1 \cdot \text{rand()} \cdot (p_{ib} - x_{id}) + c_2 \cdot \text{rand()} \cdot (p_{gd} - x_{id}) \\
  x_{id} &= x_{id} + v_{id}
\end{align*}
\]

3. FUZZY RULES GENERATION

3.1 Generating Fuzzy Rules Using FNN

The overall procedure show in Figure 2 has two parts: fuzzy modelling or training, and rule extraction. The fuzzy modelling consists of initialization and fine-tuning the fuzzy model.

The model identification procedure consists of three stages: initialization, weights learning and tuning of membership function. The last two stages are repeated until the objective function meets the stopping criterion or the number of iterations exceeds a given limit.

The rule generation is done in three steps:

- Partition of feature space: Membership functions of trained FNN divide the feature space into fuzzy regions, which contain fuzzy concepts.
- Generation of fuzzy rules: Fuzzy rules are generated from each pair of data by determining which subspace the data falls into.
- Each feature’s degrees of membership are evaluated and the feature is considered as belonging to the fuzzy set that has the maximal degree of membership.

- Significance measure of the fuzzy rules: The number of the fuzzy rules from the aforementioned steps should be the same as the number of the data pairs. This rule bank may contain conflicting and redundant rules. To resolve confliction and eliminate redundancy, the support of a rule is determined by counting the number of data that gives the same rule in each class. Then the fuzzy rules in the rule bank are ranked according to their supports.

In order to describe the problem better, we have chosen a typical model of fuzzy neural network—weighted fuzzy neural network [7]. The weighted fuzzy neural network is an adaptive network based improving fuzzy weighted reasoning method.

In the improving fuzzy weighted reasoning method, we try to list all possible fuzzy rules, the fuzzy rules have the form

\[ OR \quad AND \]

\[ j=1 \quad \text{(IF} (i=1 \quad \text{(xi IS Aij)) THEN Y IS wj1/z1, wj2/z2, \ldots, wjl/zl)) \]

where \( m \) is the number of fuzzy rules; \( n \) is the number of input variables; \( l \) is the number of consequent fuzzy set; \( zi \), \( i=1, \ldots, l \), is a constant; \( wji \), \( j=1, \ldots, m, i=1, \ldots, l \), is weight parameter; and \( Aij \), \( j=1, \ldots, m, i=1, \ldots, l \) is the antecedent fuzzy set of the \( i \)th rule. Then the output value is computed as

\[
z_0 = \frac{\sum_{j=1}^{l} \left( f \left( \sum_{j=1}^{m} u_j w_j \right) \right) \cdot z_i}{\sum_{j=1}^{l} \left( f \left( \sum_{j=1}^{m} u_j w_j \right) \right)}
\]

where \( u_j \) is the extent to which a rule is activated, it is calculated as

\[
u_j = uA1j(x1) \\text{AND} uA2j(x2) \ldots \text{AND} uAnj(xn), uAij(xi)
\]

is the membership value of \( xi \) in the \( Aij \) fuzzy set.

The weighted fuzzy neural network is a seven layer feedforward network, as shown in Fig 3.

(A): Oi = li = xi

(B): Oi = li - wbi = xi - wbi

(C): Oi = Sigmoid(wci*li) or
\[
\text{Exp}\left(-\frac{(l_i \cdot w_{ei})}{w_{ci}}\right) \text{ or } 1 - \text{Sigmoid}(w_{ci} \cdot l_i)
\]

(D) : \( O_i = \Pi_l I_i \)

(E) : \( O_i = \text{Sigmoid}(\Sigma (l_i \cdot w_{ei})) \)

(F) : \( O_1 = \Sigma w_{fi} \cdot I_i \); \( O_2 = \Sigma I_i \)

(G) : \( O = 11 / 12 \)

Figure 3. Fuzzy Neural Network Architecture

Where \( w_b \) is changeable threshold, \( w_c, w_e \) and \( w_f \) are changeable weights. The layer of \( A \) is an input layer; the layer of \( B \) and \( C \) performs fuzzification of the input data; the layer of \( D \) and \( E \) performs the fuzzy inference; the layer of \( F \) and \( G \) performs defuzzification of the output data. In this fuzzy neural network, the extraction of fuzzy rules and optimization of membership functions are how to search suitable \( w_b, w_c, w_e \) and \( w_f \). After the initialization of membership functions, use suitable algorithm on neural network to obtain improving membership functions and fuzzy rules until it achieves beforehand precision. Nodes and weights of Fuzzy neural network can be interpreted by using membership functions and fuzzy rules, thus improving membership functions and fuzzy rules can be obtained from it. The trained network can process fuzzy control and fuzzy recognition.

3.2 Generating Fuzzy Rules Using FPSO

The fuzzy systems can be also formulated as a space search problem, where each point corresponds to a fuzzy system i.e. represents membership functions, rule-base and hence the corresponding system behaviour. Given some objective/fitness function, the system performance forms a hypersurface and fuzzy model identification is equivalent to finding the optimal location on this hypersurface.

The hypersurface is generally found to be infinitely large, non differentiable and complex [7], which make evolutionary algorithms good candidate for searching the hypersurface than the traditional gradient-based methods. PSO algorithms like GAs have the capability to find optimal or near optimal solution in a given parameters of fuzzy model.

Another important consideration is the solution encoding i.e. how to represent a fuzzy system by a particle. In order for a particle to completely represent a fuzzy system, all the needed information about the rule-base and membership functions is required to be specified. It is also suggested to modify the membership functions and rule-base simultaneously, since they are code pendent in a fuzzy system.

For description of each fuzzy controller membership function introduced by the computing package, four parameters are defined. They are: \( IE \) (bellow left), \( ID \) (bellow right), \( SE \) (above left) and \( SD \) (above right). Fig. 4 shows membership function PE parameters of variable \( X \). For this function \( IE \) value is 30, \( ID \) is 160, \( SE \) is 80 and \( SD \) is 110.

For the adjustment of membership functions the following equations were defined:

\[ IE = (IE + ki) - wi \]
\[ ID = (ID + ki) + wi \]
\[ SE = (SE + ki) \]
\[ SD = (SD + ki) \]

Being \( ki \) and \( wi \) adjustment coefficient. \( ki \) makes each membership function move to the right or left with no change in the form. The membership function shrinks or expands through the factor \( wi \).

These coefficients take any integer positive or negative value, depending of the defined adjustment value made by the user.

Fig. 4 shows an adjust example of PE function with \( IE \) equal to 30, \( ID \) equal to 160, \( SE \) equal to 80 and \( SD \) equal to 110. Being \( k = -8 \) and \( w = 3 \) the membership function will have the following adjust:

\[ IE' = (30 + -8) - 3 = 19 \]
\[ ID' = (160 + -8) + 3 = 155 \]
\[ SE' = (80 + -8) = 72 \]
\[ SD' = (110 + -8) = 102 \]

PSO will be used to find the optimum values according to the strategy and the initial points used, as well as for \( ki \) and \( wi \) for the membership functions

3.2 Generating Fuzzy Rules Using FPSO

The fuzzy control usually is composed by many functions. The solution for a problem is associated with a population of particles \( p \), each particle with \( m \) dimensional, represented by a vector with \( m \) positions, \( p = \{p_1, p_2, p_3, ..., p_m\} \), where each component \( pi \) represents a \( (k, w) \) variables. Each particle represents one possible solution.

However, each particle position is composed by the adjust coefficients \( ki \) and \( wi \) which are integer values.

With regard to particle size, i.e., how many elements each particle will have, this will depend on the number of membership functions defined by user. For a fuzzy control with a group of 18 membership functions, for example, there will be a particle with 18 elements, each with 2 positions. This is because for each
function we have two adjust coefficients: \( ki \) and \( wi \). A 18 positions vector then represents one particle.

The objective of the optimization process is to find the optimal adjustment variable values.

The integration of PSO algorithms with fuzzy control was made as follow:
1. The particle was defined as a link of the membership functions adjustment values.
2. The parameters are the centres and widths of each fuzzy set. These parameters compose the particle.
3. To check the performance of the fuzzy system it is rolled up from an initial set of possible parameters.
4. This information is used for set up each particle adjustment (adaptability) and the making of the evolution of the particle.
5. The cycle repetition is made up for completion of the defined PSO iteration number made by the user. To each PSO iteration is found the best values set for the membership functions parameters.

Some of the assumptions used for the model formulation are listed below:

(i) Fixed numbers of triangular membership functions were used for both input and output variables with their centres are placed symmetrically over the universe of discourse.

(ii) First and last membership functions of each input and output variable were represented with left- and right-skewed triangles.

(iii) Complete rule-base was considered. A rule-base is considered complete when all possible combinations of input membership functions of all the input variables participate in fuzzy rule base formation.

4. CONCLUSION

The fuzzy system reflects how people think. It attempts to model our sense of words, our decision making and our common sense. As a result, it is leading to new, more human, intelligent system. Nevertheless, the system of implement by fuzzy system may become difficult for large and complex systems, when the control rules depend on subjective decisions.

This paper presented methods for generating the fuzzy rules using FNN and FPSO. Describes how it works and the differences each section. In this FNN, the extraction of fuzzy rules and optimization of membership function are how to search suitable weight. After the initialization of membership functions, we use suitable algorithm on neural network to obtain improving membership functions and fuzzy rules until it achieves beforehand precision. Nodes and weights of Fuzzy neural network can be interpreted by using membership functions and fuzzy rules, thus improving membership functions and fuzzy rules can be obtained from it.

PSO algorithm is applied by fitting fuzzy membership functions. The PSO training is added with an automatic technique for the fitting of the membership functions parameters. PSO is able to generate an optimal set of parameters for fuzzy reasoning model based on either their initial subjective selection or on a random selection. It is also shown that by training this algorithm, with some specific initialization, it has reached a good global optimization result. This technique is presented as a great promise for optimization process.

For future works, we can see that FNN has training data with Neural Network and only use weight parameter to optimize membership functions. The other hand FPSO uses training data with some specific initialization and has reach good optimization result. It will be interesting if the two methods mix together. NN is used for training and PSO will used for optimization.

5. ACKNOWLEDGMENTS

This paper is supported by RMC UTM

6. REFERENCES


