# Fuzzy systems, neural networks and genetic algorithms

Hyung Lee-Kwang, Jee-Hyong Lee

Department of Computer Science/Advanced Information Technology Research Center,
KAIST(Korea Advanced Institute of Science and Technology),
373-1, Kusong-dong, Yusong-gu, Taejon 305-701, Korea
Email:{khlee,jhlee}@fuzzy.kaist.ac.kr

## Abstract

Fuzzy systems, neural networks and genetic algorithms have different origins and thus have differently developed their own unique characteristics. These characteristics can be used as a good complement to the others. Therefore, many researches have been devoted to not only these techniques but also fusion of them. This paper briefly summarizes these three techniques and surveys the researches on fusion of them.

## 1 Introduction

Fuzzy systems, neural networks and genetic algorithms were proposed around the middle of this century. Since then these techniques have been developed and applied to many application areas.

Fuzzy systems were proposed to represent and manage vague knowledge, neural networks were developed as a computational model which had a learning capability, and genetic algorithms as a heuristic search method. These different characteristics are good complements to each other. For example, neural networks and genetic algorithms may give fuzzy systems a learning algorithm and fuzzy systems may help to present priori knowledge to neural networks and genetic algorithms. Thus, since late 1980s research on fusion of them has been performed. In 1990s the amount of research on these technique and fusion of them has rapidly increased. This paper briefly summarizes fuzzy systems, neural networks, genetic algorithms and researches on fusion of them.

## 2 Fuzzy systems, neural networks and genetic algorithms

### 2.1 Fuzzy systems

Fuzzy logic is a powerful problem-solving methodology in wide applications including control, operations research, pattern recognition, database and expert systems. For example, fuzzy control has been actively applied and given satisfactory results in the industrial fields judging by the billions of dollars worth of sales and close to 2000 patents issued in Japan alone.

Fuzzy systems provides an alternative way of thinking, which allows to model complex systems using a higher level of abstraction originating from our knowledge and experience. In fuzzy systems, complex systems are described using expert knowledge and experience in simple high level language like rules. It does not require any system modeling or complex math equations governing the relationship between inputs and outputs.

Generally, fuzzy systems have fuzzy rules which describe specific knowledge such as how to control a system. Fuzzy rules are IF-THEN rules whose antecedent and consequent parts are linguistically described, i.e., described with linguistic terms such as "mall", "Medium", etc. The followings are examples of fuzzy rules:

- `if` $x$ is Small `then` $y$ is Large
- `if` $x_1$ is Very Large `and` $x_2$ is Medium `then` $y$ is Very Small

That is, fuzzy rules are a set of simple input-output relations. Using these fuzzy rules, fuzzy systems can model a complex nonlinear input-output relation.

Figure 1 shows the general structure of fuzzy systems. The fuzzification interface fuzzyfies input values. It converts input values into linguistic terms. The knowledge base consists of the database and the fuzzy rule base. In the database, there are definitions which need for manipulation of fuzzy data and definition of fuzzy rules. The fuzzy rule base contains the linguistic fuzzy rules. The inference engine performs inference and generates the output, and the defuzzification interface defuzzifies the output.

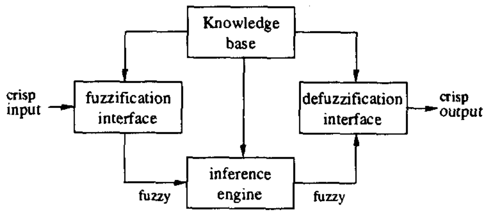


Figure 1: Structure of fuzzy systems

## 2.2 Neural networks

Neural networks are a computational model of the operation of the brain. A neural network is composed of a number of nodes, or units, connected by links. Each link has a numeric weight associated with it. Weights are the primary means of long-term storage in neural networks. One of the major features of a neural network is its learning capability. They can adjust the parameters in a neural network such that the network learns to improve its performance for a given task. Learning usually takes place by updating the weights.

A unit has a set of input links from other units, a set of output links and output links and a nonlinear switching function. The functions of a unit are 1)summation of inputs multiplied by the weight of the corresponding link and 2)generation of output using the switching function and the output of 1).

There are two types of neural network learning algorithms. The first type is supervised learning. It uses a set of training data which consist of pairs of input and output. During learning, the parameters of a neural network are changed so that the input-output mapping of the neural network becomes more and more close to the training data. There are many neural network architectures which use supervised learning, such as multilayer feed-forward neural networks, Adaline, hopfield network, etc. Among these, multilayer feed-forward neural network and its learning algorithm, error backpropagation are widely used. The second type is unsupervised learning. While supervised learning presents target answers for each input to a neural network, unsupervised learning is not given the target answers. It adjusts the weights of a neural network in response to only input patterns without target answers. In unsupervised learning, the network classifies the input patterns into similarity categories.

## 2.3 Genetic algorithms

Genetic algorithms can be viewed as a general-purpose search method, an optimization method, or a learning mechanism, based loosely on Darwinian principles of biological evolution: reproduction and "survival of the fittest" along with genetic recombination. Genetic algorithms maintain a set of candidate solutions called a population. Candidate solutions are usually represented as strings of fixed length, called chromosomes.

Given a initial population, genetic algorithms operate in cycles, called generations, as follows:

1. Initialize a population of chromosomes.
2. evaluate each chromosome in the population.
3. Create new chromosomes by mating current chromosomes; apply genetic operators.
4. Delete some chromosomes of the population to make room for the new chromosomes.
5. Evaluate the new chromosomes and insert them into the population.
6. If time is up, stop and return the best chromosome; if not, go to 3.

There are crossover operator and mutation operator as representative genetic operators. The crossover operator produces two new offsprings(candidate solutions) by recombining the information from two parents(existing candidate solutions). The mutation operator prevents irreversible loss of patterns by introducing small random changes into chromosomes. It has been shown that mutation plays a decidedly secondary role in the operation of genetic algorithms.

A number of factors have an effect on the operation of genetic algorithms: the size of the population, the size of the subpopulation replaced in each cycle, the probability of applying the genetic operations, etc. Thus, we need to carefully choose these values.

# 3 Fusion of fuzzy systems and neural networks

Neural networks and fuzzy systems are two complimentary technologies. Neural networks can learn from data, but the knowledge represented by the neural networks is difficult to understand. In contrast, fuzzy systems are easy to comprehend because it uses linguistic terms and if-then rules, but it does not have learning algorithms. So, many researches have been devoted to fusion of both. The researches on fusion of neural networks and fuzzy systems can be classified into four categories:

- modifying fuzzy systems with supervised neural network learning,
- making membership functions with neural networks,
- building neural networks on fuzzy partitioned input space,
- concatenating neural networks and fuzzy systems.

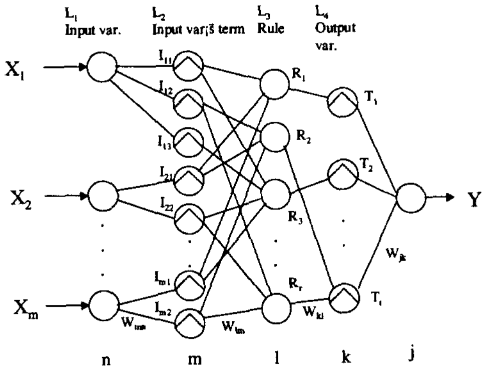


Figure 2: Fuzzy neural networks

## 3.1 modifying fuzzy systems with supervised neural network learning

The proposed fuzzy neural network is a multilayer feedforward neural network. This neural network represents fuzzy rules and performs inference with given inputs. In a multilayer feedforward neural network, nodes have the same functionality and are fully connected to the nodes in the neighboring layers, but in a fuzzy neural network, nodes have different functionalities and are not fully connected to nodes in the neighboring layers. This differences come from the fact that the nodes and links in a fuzzy neural network usually correspond to a specific component in a fuzzy system. That is, some nodes represent the linguistic terms of input variables, some nodes are for those of output variables, and some nodes and links are used for representing fuzzy rules.

For example, the fuzzy neural network proposed by Kwak[8] consists of five layers as shown in Figure 2. The functions of nodes in each layer are as follows:

*First layer*: The nodes in the first layer takes inputs and just pass those inputs to the second layer.

*Second layer*: A node in this layer represents a linguistic term of an input variable. It has parameters which represent the membership function of a linguistic term. For example, in Figure 2 the input variable $X_1$ is connected to three nodes in the second layer and $X_2$ is connected to two nodes. It means that there are three linguistic terms defined on $X_1$ and two linguistic term on $X_2$. The node outputs the membership degree of a input to the linguistic term it represents.

*Third layer*: This layer corresponds to the antecedent parts of fuzzy rules. For example, the inputs of $R_1$ are the outputs of $I_{11}$, $I_{21}$ and $I_{m1}$. It represents "`if` $X_1$ `is` $I_{11}$ `and` $X_2$ `is` $I_{21}$ `and` $X_m$ `is` $I_{m1}$ `then`". The output of a node is the matching degree of given inputs to the antecedent part of a rule.

*Fourth layer*: This layer represents the consequent parts of fuzzy rules. Likely a node in the second layer, a node of this layer represents a linguistic terms of the output variable. For example, the node $T_t$ has two inputs from $R_2$ and $R_r$. It represents two rules: "`if` the antecedent part is $R_2$ `then` $Y$ is $T_t$" and "`if` the antecedent part is $R_r$ `then` $Y$ is $T_t$".

*Fifth layer*: A node in this layer gathers the outputs of all rules and defuzzifies them.

The learning algorithm of this model is based on error backpropagation. During learning process, the weights of links between the third and the fourth layer and the parameters representing membership functions in the nodes of the second and the fourth layer are modified.

## 3.2 making membership functions with neural networks

An important element of fuzzy systems is the fuzzy partition of the input space. So, if there are $k$ inputs, the fuzzy rules generates $k$-dimensional fuzzy hypercubes in the input space. Even though we can understand these fuzzy hypercubes, they are not so flexible for partitioning nonlinear hypersurfaces in the input space. Since neural networks are proper to approximate nonlinear functions, they can be applied to constructing a fuzzy partition.

This approach is useful if the input space can be partitioned into several classes by fuzzy hypersurfaces. The idea of this model is building a neural network whose outputs are degrees that an input belongs to each class. This degrees can be considered as the membership degrees to each class. That is, the neural network takes the role of membership functions. Thus the membership functions of this model can be non-linear and multi-dimensional unlike conventional fuzzy systems.

This model has been used in the control of rolling mill [7]. The purpose of the rolling mill is to make plates of iron, stainless, or aluminum by controlling 20 rolls. The controller will change system parameters if the surface of plate is not flat. How to change the parameters is determined by the produced surface shape of plates. To do this, first fuzzy rules are generated for only 20 standard template surface shapes. Then, a neural network is constructed which generates the similarity degrees to which an arbitrary surface shape belongs to each standard template shape. The similarity degrees produced by the neural network are used as the matching degrees to the antecedent part of each rule. Since the antecedent parts of fuzzy control rules are standard template surface patterns, the output of the neural network corresponds to how much input surface pattern matches to each fuzzy rule.

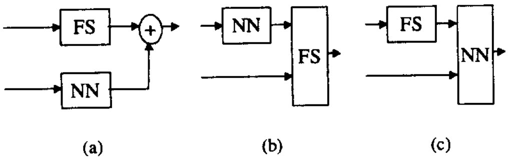


Figure 3: Combinations of the concatenated systems of neural networks and fuzzy systems

### 3.3 building neural networks on fuzzy partitioned input space

In the above two categories, neural networks are used for designing or improving the performance of fuzzy systems, but the methods in this category use fuzzy rules for designing neural networks. This model is a kind of divide and conquer approach. Instead of training a neural network for the whole given input-output data, this model

1. divides the given input-output data into several classes with fuzzy rules,
2. builds a neural network per class, and
3. trains the neural networks with the input-output data in the corresponding class.

The complexity in each fuzzy partitioned input space is much less than that of whole given task.

This approach is suitable to the problems which can be divided into small sub-problems. For example, recognition of characters of multiple fonts can be divided into character recognitions of each font. If there are fuzzy rules which identify the font of a character, we can classify characters with the fuzzy rules according to fonts and then recognize the characters with the recognizer specialized to the corresponding font. It must be easier to make several recognizers which identify characters of a font than a recognizer which identifies those of several fonts.

With this method, we can use a priori knowledge of the task into the system to reduce the complexity of a given task and increase the performance of a system. The fuzzy rules describe a priori knowledge of the given task. They can be obtained by analyzing training data.

### 3.4 concatenating neural networks and fuzzy systems

This category includes the methods equally using fuzzy systems and neural networks for improving system performance. The first is the correction of the output of an fuzzy system with the output of a neural network to increase the precision of the final system output. Figure 3(a) shows the combination of a neural network and fuzzy system in this approach. If an fuzzy system exists and input-output data set is available, this model can be used for improving the performance without modifying the existing fuzzy systems. The second is cascade combination of a neural network and an fuzzy system where the output of the fuzzy system or neural network becomes the input of another neural network or fuzzy system. Figure 3(b) shows the model where the output of an fuzzy system is inputted to a neural network and Figure 3(c) shows the reverse model. An example of Figure 3(b) is controlling oven ranges, a neural network first estimates the number of pieces of bread from sensor information, and then an fuzzy system determines the cooking time and the power by using the output of the neural network(the number of pieces) and other sensor information.

## 4 Fusion of fuzzy systems and genetic algorithms

Like fuzzy systems and neural networks, fuzzy systems and genetic algorithms can complement each other very well. Researches on fusion of fuzzy systems and genetic algorithms can be classified into two categories: identifying fuzzy systems with genetic algorithms and controlling parameters of genetic algorithms with fuzzy systems.

### 4.1 identifying fuzzy systems with genetic algorithms

Although fuzzy systems have been used to control a number of systems, the selection of acceptable fuzzy membership functions has been a subjective and time-consuming task. When we build a fuzzy system, we should determine the number of the linguistic terms of input and output variables, their membership functions and the consequence parts of fuzzy rules. The if-then structure of fuzzy rule are easy to understand and to build with priori knowledge, but many parameters of fuzzy rules should be specified by an expert.

The identification of these parameters in a fuzzy system can be viewed as an optimization problem, finding parameter values that optimize the model based on given evaluation criteria. Therefore, there have been many researches on applying genetic algorithms to the identification of fuzzy systems. The researches can be categorized into groups: 1)tuning a fuzzy system and 2) building a fuzzy system.

The researches in the first category modify the parameters of an existing or given fuzzy system. The usually tuned parameters are the membership functions of linguistic terms and fuzzy rules. Tuning the membership functions with genetic algorithms are analogue to fuzzy neural networks. In these researches, the membership functions are encoded into chromosomes and better membership functions are searched by genetic algorithms. In order to tune fuzzy rules, the consequent part of fuzzy rules are encoded. For example, there are four fuzzy rules: `if` $X$ `is` $I_1$ `then`

$Y$ is $O_1$, if $X$ is $I_2$ then $Y$ is $O_2$, if $X$ is $I_3$ then $Y$ is $O_3$, and if $X$ is $I_4$ then $Y$ is $O_4$. Then, these are encoded as a string of linguistic terms like $O_1O_2O_3O_4$. The genetic operators will change linguistic terms themselves, not the membership functions of them, for example $O_1O_2O_3O_4$ may be changed into $O_1O_3O_4O_1$ after genetic operations. This approach is proper to rough tuning of fuzzy systems because changing rules may affect the fuzzy system much.

The methods tuning fuzzy systems assume that a fuzzy systems is available; it may be an existing one or roughly built by an expert. However, the methods building a fuzzy system do not need such a fuzzy system. They determine all parameters of a fuzzy system by genetic algorithms with out any priori knowledge . Thus, the chromosomes used in these method usually include most of fuzzy system parameters such as the number and membership functions of linguistic terms, and fuzzy rules. So, it is very important how to effectively represent those parameters because long chromosomes means a wide search space. If a search space is wide, we cannot expect a good optimization result. So, most researches make restrictions; for example, some fix the number of linguistic terms or restrict the shape and position of membership functions. In a sense that this approach does not use any priori knowledge, it is analogue to neural networks.

### 4.2 controlling parameters of genetic algorithms with fuzzy systems

Genetic algorithms need some parameters such as population size, and crossover and mutation rates. These parameters are very important to the performance of genetic algorithms, and the interaction between them is known complex. Thus, there have been many researches on how these parameters effect on genetic algorithms performance and how to set them to improve the performance. However, the parameters setting is often left to the user and never changed during evolutions.

There are some research on controlling dynamically parameters of genetic algorithms during evolutions with a fuzzy system. The basic idea is simple; a fuzzy system observes the states of population during evolutions and changes the parameters to improve the performance. That is, genetic algorithms uses a fuzzy knowledge-based system to dynamically control the parameters, such as population size, crossover rates, and mutation rates. For example, the fuzzy rules can be described as follows:

- if average fitness is high then population size should be increased.
- if best fitness is not improved then mutation rate should be increased.

One question of this approach is how to obtain the knowledge to build fuzzy rules. It can be solved in several ways; an expert on genetic algorithms can describe his/her own knowledge or an automatic fuzzy design technique can be applied.

## 5 Fusion of genetic algorithms and neural networks

Likely fusion of neural networks and fuzzy systems, or fuzzy systems and genetic algorithms, genetic algorithms and neural networks may compliment each other. One of research trend of fusion of neural networks and genetic algorithms is neural network based fitness function for genetic algorithms. It uses the learning capability of neural networks for devising a fitness function of genetic algorithms. Another trend is training or configuring a neural network with genetic algorithms.

### 5.1 evaluating chromosomes with neural networks

In a genetic evolutionary process, every chromosomes are evaluated to verify how much each of them are fit to the given problems. For this purpose, fitness functions are used. Thus, we have to devise a mathematical function which can evaluate chromosomes. However, in many cases we hardly find such functions for evaluation, Most of these cases are finding parameters which can optimize some process. In the cases optimizing a process, almost only way to evaluate parameters in a chromosome is applying those parameters to the real process and observing what happens in the process. However, the cost of this approach must be very high. There are hundreds of chromosomes and they evolve through hundreds to thousands of generations, so we have to run the real process with tens or hundreds of thousands of different parameters.

One of this solutions is to make a simulator of the given process and devise a fitness function using the last state or the output of a simulated process. Neural network may be a useful tool to build a simulator because it is not so difficult to observe the given process and get input-output data.

An example of this approach is finding time patterns of water drainage and supply for a hydroponics systems [3]. The hydroponics system controls the time pattern of water drainage and supply to the target plant to maximize its photosynthetic rate. For this, a neural network simulating growth of the target plant is built, and time patterns and the amount of $CO_2$ turned out by the plant are used as the training data. Then, time patterns of water drainage and supply are generated by the genetic algorithm and applied to the plant simulator. The best time pattern selected with the genetic algorithm and the simulation is applied to the actual plant.

### 5.2 training and configuring neural networks using genetic algorithms

Another research trend on fusion of genetic algorithms and neural networks is training or configuring a neural network using genetic algorithms.

Training a neural network is modifying the weights of links, so we can think of the training as a problem searching the best weights. So, genetic algorithms can be applied to train a neural network instead of learning algorithms. In this case, the weights are encoded into chromosomes and the fitness of a chromosome may be how much a neural network with the weights in a chromosome makes output close to the target values.

When we build a neural network, we first determine how many layers will be in the neural network and how may nodes will be in each layer. This determination is generally very subjective; a user determines these with his/her experience and some heuristic guide lines. For automatic determination of these, genetic algorithms may be used. The numbers of layers and nodes are encoded into chromosomes. In order to evaluate a chromosome, we first construct a neural network with as many layers and nodes as specified in the chromosomes, train the neural network with the given input-output data, and evaluate how much the neural network outputs closely to the target value. One defect of this approach is that the evaluation may take too much time.

## 6 Concluding remarks

Fuzzy systems, neural networks and genetic algorithms have been developed since the middle of this century. Since each of them has its own unique characteristic, these techniques had independently been developed and applied till early 1980s. However, these days those different characteristics have been used as complements to each other. This paper summarized fuzzy systems, neural networks, and genetic algorithms, and surveyed researches on fusion of them.

## References


[1] H. Bersini, G. Scront, "In search of a good evolution-optimization crossover," in Parallel Problem Solving from Nature, R. Manner and B. Mandrick(eds.), Elsevier Science Publishers, pp.479-488, 1992.

[2] C. Karr, L. Freeman, D. Meredith, "Improved Fuzzy Process Control of Spacecraft Autonomous Rendezvous Using a Genetic Algorithm," SPIE Conf. on Intelligent Control and Adaptive Systems, pp.274-283, 1989.

[3] T. Morimoto, T. Takuchi, Y. Hashimoto, "Growth optimization of plant by means of the hybrid system of genetic algorithm and neural network," Int'l Joint Conf. on Neural Networks, Nagoya, Japan, Vol.3, pp.2979-2982, 1993.

[4] T. Tokagi, M. Sugeno, "Fuzzy Identification to Systems and Its Applications to Modeling and Control," IEEE Trans. System, Man, Cybern., Vol.15, pp.116-132, 1985.

[5] H. Takagi, N. Suzuki, T. Kouda, Y. Kojima, "Neural networks designed on Approximate Reasoning Architecture," IEEE Trans. Neural Networks, Vol.3, pp.752-760, 1992.

[6] H. Takagi, "Industrial and Commercial Applications of NN/FS/GA/Chaos in 1990s." Int'l Workshop on Soft Computing in Industry (IWSCT'96), Muroran, Hokkaido, Japan, pp.160-165, 1996.

[7] H. Takagi, "Introduction of fuzzy systems, neural networks, and genetic algorithms," Intelligent Algorithms: Fuzzy Logic, Neural Networks and Genetic Algorithms, D. Ruan(eds.), Kluwer Academic Publishers, 1997.

[8] D. H. Kwak, "Learning of Fuzzy Rules by Fuzzy Neural Networks," M. S. Dissertation, Department of Computer Science, KAIST, 1994(in Korean).

[9] K. M. Lee, D.H. Kwak, H. Lee-Kwang, "Tuning of Fuzzy Models by Fuzzy Neural Networks, Fuzzy Sets and Systems, Vol. 76, pp.47 - 61, 1995.

[10] K. M. Lee, D. H. Kwak, H. Lee-Kwang, "Fuzzy Inference Neural Network for Fuzzy Model Tuning," IEEE Trans. Systems, Man and Cybernetics, Vol. 26, No. 4, pp.637 - 645, 1996.

[11] J. H. Lee and H. Lee-Kwang, "Fuzzy Identification of Unknown Systems based on GA," Lecture Notes in Artificial Intelligence, X. Yao, J. H. Kim, T. Furuhash(eds.), Springer, Vol. 1285, pp.216 - 223, 1997.

[12] H. Lee-Kwang, K. M. Lee, "Techniques in fuzzy inference neural networks for fuzzy model provement and their application," Fuzzy Theory Systems Techniques and Applications, 1998.

[13] C. C. Wong, C. S. Fan, "Rule mapping fuzzy controller design," Fuzzy Sets and Systems, Vol.108, pp.253-261, 1999.

[14] S. B. Cho, "Pattern recognition with neural networks combined by genetic algorithm," Fuzzy Sets and Systems, Vol.103, pp.339-347, 1999.

[15] K. M. Lee, H. Leekwang, "Identification of $\lambda$-fuzzy measure by genetic algorithms," Fuzzy Sets and Systems, Vol.75, pp.301-309, 1995.