Optimization of Neuro-Fuzzy System Using Genetic Algorithm for Chromosome Classification

M. Sarosa,[1,2] A.S. Ahmad,[1] B. Riyanto[1] & A.S. Noer[3]

[1]School of Electrical Engineering and Informatics, Institute of Technology Bandung,
[2] Departement of Electrical Engineering, Polytechnic of Malang, Indonesia
[3] Division of Biochemistry, Institute of Technology Bandung, Indonesia

Abstract. Neuro-fuzzy system has been shown to provide a good performance on chromosome classification but does not offer a simple method to obtain the accurate parameter values required to yield the best recognition rate. This paper presents a neuro-fuzzy system where its parameters can be automatically adjusted using genetic algorithms. The approach combines the advantages of fuzzy logic theory, neural networks, and genetic algorithms. The structure consists of a four layer feed-forward neural network that uses a GBell membership function as the output function. The proposed methodology has been applied and tested on banded chromosome classification from the Copenhagen Chromosome Database. Simulation result showed that the proposed neuro-fuzzy system optimized by genetic algorithms offers advantages in setting the parameter values, improves the recognition rate significantly and decreases the training/testing time which makes genetic neuro-fuzzy system suitable for chromosome classification.

Keywords: *chromosome classification; genetic algorithms; Neuro-fuzzy system; optimization; recognition rate.*

1 Introduction

The use of Pattern Recognition and Image Analysis is becoming increasingly popular in many fields of Biology and Medicine. One of these applications is the classification of human chromosomes. Chromosome analysis is important in many situations such as prenatal amniocentesis examination, detection of malignant diseases such as leukemia, gene mutations, and monitoring environmental. The normal human "karyotype" or set of chromosomes consists of 22 homologous pairs or "autosomas" and one pair of sex chromosomes [1].

Chromosome classification and analysis are aided by the use of automated karyotyping systems that yield a preliminary classification for each chromosome, which may be corrected manually as necessary. Automated karyotyping relies upon acquisition of a digital image, followed by extraction of chromosome features. Two general approaches feature extraction are employed: gray level encoding of each chromosome and more complex extraction of

Received May 20th, 2007, Revised August 16h, 2007, Accepted for publication August 16th, 2007.

distinctive features. These features may then be used in an algorithm that assigns the chromosome to one of 24 classes (autosomes 1–22, X, and Y) [2].

A variety of such algorithms have been proposed, based upon approaches such as Bayesian analysis [2], Markov Networks [3], Neural Networks [4], Multilayer Percentron [1], k-Nearest Neighbours Search Technique [5], Hidden Markov Models [6], Maximum Likelihood Techniques [7]. The reported classification accuracy by using these approaches varies insignificantly. Most methods achieve approximately 90% correct classification of the Copenhagen chromosome data set; commercial implementations typically achieve approximately 80% correct classification in routine use [6].

The neuro-fuzzy system (NFS) that will be optimized is a four feed-forward neural network. This system uses four parameters which are adjusted to find an optimum recognition result. It is not easy and needs a long time to obtain the accurate the four parameter values [8]. The genetic algorithms (GAs) as a random search technique are proposed to optimize the NFS. The NFS with GAs, called the Genetic Neuro-Fuzzy System (GNFS), can obtain the precise parameters values necessary to provide the best recognition result.

2 Neuro-Fuzzy System

Artificial neural networks and fuzzy inference system are both very powerful soft computing tools for solving a problem without having to analyze the problem itself in detail [9]. The architecture of NFS is a modification of a four layer feedforward neural network presented is [10]. This architecture has been shown to provide good performance on chromosome classification [8].

2.1 Structure of The NFS

The NFS structure is shown in Figure 1. The first layer is the input layer which accepts patterns into network. Assuming that each input pattern has *N* numbers, then the first layer has *N* INPUT-FN's. The algorithm of the i^{th} INPUT-FN in the first layer is shown in Equations 1 and 2.

$$s_i^{[1]} = z_i^{[1]} = x_i, \quad for\ \ i = 1\ to\ N \tag{1}$$

$$y_i^{[1]} = \frac{s_i^{[1]}}{P_{v\,max}} = \frac{x_i}{P_{v\,max}}, \ for\ \ i = 1\ to\ N \tag{2}$$

where x_i is the *i*th value of an input pattern ($x_i \geq 0$), and P_{vmax} is the maximum value among all input patterns.

The purpose of this layer is to fuzzify input patterns through a weight function $w[n]$ and is called the fuzzification function. The state of the n^{th} MAX-FN in this layer is presented in Equation 3.

$$s_p^{[2]} = \max_{i=1}^{N} \left(w[p-i] y_i^{[1]} \right) \quad \text{for } p = 1 \text{ to } N. \tag{3}$$

where $w[p\text{-}i]$ is the weight connecting the ith INPUT-FN in the first layer to the p MAX-FN in the second layer defined by Equation 4.

$$w[m] = \exp\left(-\beta^2 m^2\right) \quad \text{for } m = -(N-1) \text{ to } (N-1). \tag{4}$$

The value of β is decided by the learning algorithm.

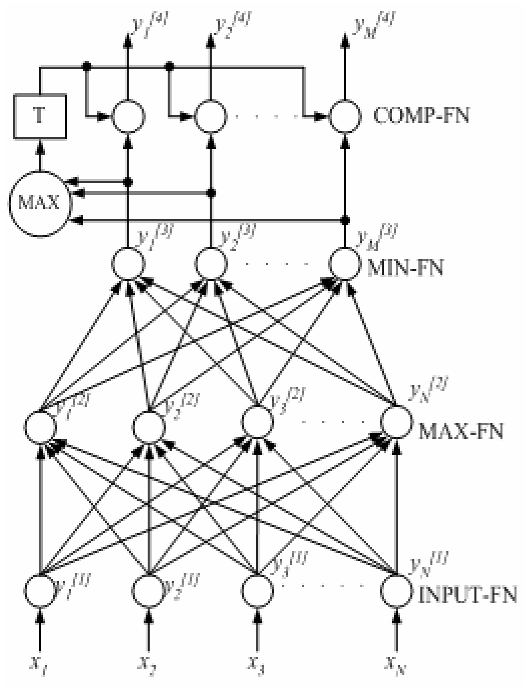

Figure 1 Four-layer feedfoward NFS.

Each MAX-FN in this layer has M different outputs (M is the number of FN's in the third layer), one for each FN in the third layer. The outputs pth MAX-FN in this layer are shown in Equation 5.

$$y_{pm}^{[2]} = g_{pm}\left[s_p^{[2]}\right] \tag{5}$$
for $p = 1$ to N, $m = 1$ to M.

where $y_{pm}^{[2]}$ is the mth output of the pth MAX-FN which is connected to the mth MIN-FN in the third layer. The output function $g_{pm}\left[s_p^{[2]}\right]$ is to be determined by the learning algorithm. The Generalized Bell (GBell) membership function is used as the output function of the MAX-FN's in the second layer, as shown in Figure 2 [11].

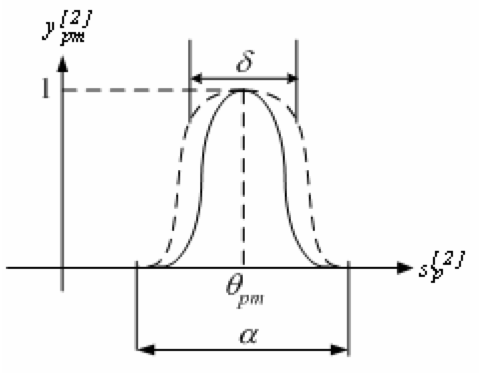

Figure 2 Output function of a MAX-FN in the second layer.

Assuming that α is base wide, δ is bell culminate wide, and θ_{pm} is central point of the base of function $g_{pm}\left[s_p^{[2]}\right]$, the output of second layer is presented in Equation 6 [12].

$$y_{pm}^{[2]} = g_{pm}\left[s_p^{[2]}\right] = \begin{cases} \dfrac{1}{1 - \left|\dfrac{\left|s_p^{[2]} - \theta_{pm}\right|}{\alpha}\right|^{2\delta}} & \text{if } \left|s_p^{[2]} - \theta_{pm}\right| \neq 0 \\ 1 & \text{if otherwise} \end{cases} \tag{6}$$
for $p = 1$ to N, $m = 1$ to M.

The determination of α, δ, and θpm for every set of p and m are provided by learning algorithm discussed in Section 2.2.

Each MIN-FN in the third layer represents one learned pattern. Hence, the number of MIN-FN's in the third layer, M, could be determined only after the learning procedure is finished. The output of the mth MIN-FN in the third layer is:

$$y_m^{[3]} = s_m^{[3]} = \min_{p=1}^{N}\left(y_{pm}^{[2]}\right) \qquad m = 1 \text{ to } M \tag{7}$$

where $s_m^{[3]}$ represents the state of the mth MIN-FN in the third layer.

COMP-FN's in the fourth layer are nodes in the output layer, one for each of the M learned patterns, and providing nonfuzzy outputs. If an input pattern is most similar to mth learned pattern, then the output of the mth COMP-FN in the fourth layer is 1 while other outputs are 0. The number of COMP-FN's in the output layer is equal to M. The algorithm of the mth COMP-FN in the fourth layer is defined in Equations 8, 9 and 10.

$$s_m^{[4]} = z_m^{[4]} = y_m^{[3]} \quad for\ m = 1\ to\ M \tag{8}$$

$$y_m^{[4]} = g\left[s_m^{[4]} - T\right] = \begin{bmatrix} 0 & \text{if } s_m^{[4]} < T \\ 1 & \text{if } s_m^{[4]} = T \end{bmatrix} \qquad \text{for } m = 1 \text{ to } M. \tag{9}$$

$$T = \max_{m=1}^{M}\left(y_m^{[3]}\right) \quad for\ \ m = 1\ to\ M \tag{10}$$

where T is the activation threshold of all the COMP-FN's in the fourth layer.

2.2 Self-Organizing Learning Algorithm

The following parameters must be determined by the learning procedure. The parameters of output functions of the MAX-FN's in the second layer, α, δ, and θ_{pm} (for each set p and m), the parameter of the fuzzufication function, β, and the number of FN's in each of the third and fourth layers, M. We define T_f as the fault tolerance of the FNN ($0 \leq T_f \leq 1$) and K as the total number of training patterns (for k=1 to K).

Step 1: Create N INPUT-FN's in the first layer and MAX-FN's in the second layer. Choose a value for $\alpha\,(\alpha \geq 0)$, $\delta\,(\delta \geq 0)$ and a value for β.

Step 2: Set M=0 and k=1.

Step 3: Set M=M+1. Create the Mth MIN-FN in the third layer and the Mth COMP-FN in the fourth layer. Set Equation 11.

$$\theta_{pM} = S_p^{[2]} = \max_{i=1}^{N} \left(w[p-i]\, X_{ik} \right) \tag{11}$$

for p=1 to N
where θpM is the central point of the Mth output function of the pth MAX-FN in the second layer. Xk={xik} is kth training pattern.

Step 4: Set k=k+1. If k>K then the learning procedure finished. Otherwise, input the kth training pattern to the network and compute the output of the current NFS (with M FN's in the third and fourth layers). Set Equation 12.

$$\sigma = 1 - \max_{j=1}^{M} (y_{jk}^{[3]}) \tag{12}$$

where $y_{jk}^{[3]}$ is the output of the jth MIN-FN in the third layer for the kth training pattern Xk. If $\sigma \leq Tf$, go to Step 4. If $\sigma > Tf$, go to Step 3.

3 Genetic Algorithms

GAs, first proposed by Holland in 1975 [13], is a derivative-free stochastic optimization approach based on the concept of biological evolutionary processes. GAs encodes each point in solution space into binary bit string called chromosome, and then evaluates each chromosome by a fitness function. Such a fitness function corresponds to the objective function of the original problem. Usually, GAs keeps a pool of chromosomes called population at the same time and these chromosomes can be evolved by selection, crossover, and mutation operator. Further discussion on GAs can be obtained in Goldberg in 1989 [14], Attia in 2001 [15], and Cordon in 2004 [16]. Sexton *et. al.* [17], have recently demonstrated that the genetic algorithm appears to be able to systematically obtain more superior solutions than simulated annealing for optimizing neural networks.

To implement GAs as a learning procedure for the NFS, the parameters are coded into string of chromosome. For each instances in the population of chromosomes, the genetic operator such as crossover with probability rate P_c and mutation with probability rate P_m are performed by Topchy in 1996 [18].

The fitness is proportional to the whole recognition rate and system error of learning procedure. After a number of generations, the population will contain, hopefully, chromosomes with better fitness values. The population will evolve until a stop criterion is satisfied, and among which, the chromosomes with lowest fitness values is chosen as a best solution for the original problem [19].

To describe the GAs optimization process, consider the functional block diagram shown in Figure 3. At the beginning of the process, the initial populations comprise a set of chromosomes that are scattered over the search space. The initial population may be randomly generated.

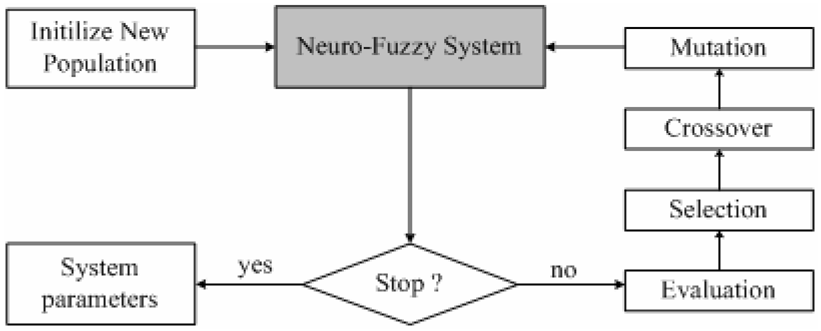

Figure 3 A functional block diagram showing the GA optimization process.

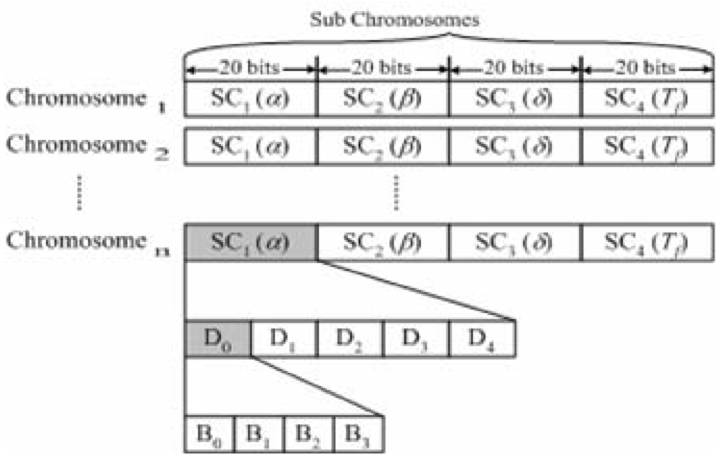

Figure 4 Chromosomes Format.

In this paper, we devide each chromosomes into 4 sub-chromosomes that represent the system parameters (α, β, δ, Tf). To facilitate the encoding process of chromosome value, the binary to code decimal (BCD) is used in each chromsosomes. The sub-chromosome has 5 digits BCD format, and therefore the precision value is 10^{-5}. The BCD requires 4 bits so that the bit number of chromosome is 4x5x4 = 80 bits. Because the digit number of GNFS parameter values is higher than that of NFS, the parameter obtained by genetics algorithm turn out to be more precise. Figure 4 shows the chromosome format.

NFS training can be applied using parameter (α, β, δ, dan Tf) values produced from sub chromosome bits encoding. This process will adaptively build the layer 3 and 4 of neuron network which in the end of the training will produce recognition error rate. The fitness value is expressed as the number of neurons produced (M) and the number of pattern trained (N), which used to select next generation chromosomes. The reproduction process will terminate when the fitness value is sufficiently small and the recognition error rate is below the allowed value. At this point, the system is able to choose unsimilar patterns which will serve as a reference at testing process.

Reproduction process consists of evaluation, selection, crossover, and mutation. Evaluation is a jugdement process to filter chromosomes by their fitness values. Selection will choose the chromosome to keep at next generation. In this research, we used principle of Roulette Wheel to make that choice [14]. While crossover or recombination represents the process of sub chromosome bits transfer from two chosen chromosome randomly, mutation is the process of bits replacement at sub chromosome in which its position and value are determined randomly.

4 Dataset

The data used in this paper was extracted from a database of approximately 7,000 chromosomes that are classified by cytogenetic experts [2]. Each digitized chromosome image was automatically transformed into a string through a procedure that obtaining an idealized, one dimensional density profile that emphasizes the band pattern along the chromosome [5]. A total of 4400 samples were collected, consisting of 200 samples of each of the 22 non-sex chromosome types. See [3] for details about this preprocessing.

5 Experimental Result and Analysis

We have simulated the proposed GNFS on P4-PC (2.13 GHz) using Delphi language. As listed in Table 1, 4400 patterns of Copenhagen Chromosome Database are grouped in 5 sets of training data and 5 sets of testing data.

Table 1 Five sets of training patterns and testing patterns.

Type	Number of Training patterns	Type	Number of Testing patterns
Train 1	220	Test 1	220
Train 2	550	Test 2	550
Train 3	1100	Test 3	1100
Train 4	1650	Test 4	1650
Train 5	2200	Test 5	2200

The result of the fitness variation chromosome reproduction is presented in Figure 5. From the graph, it can be seen that chromosome reproduction has influenced the fitness value from generation to generation. Because the fitness value is the ratio of the number of neurons in 3rd and 4th layer and the number of input pattern, the degradation of the fitness value indicates that the system have been able to choose the patterns which do not have similarity. The smaller fitness value indicates that system has ability to select the patterns with similarities.

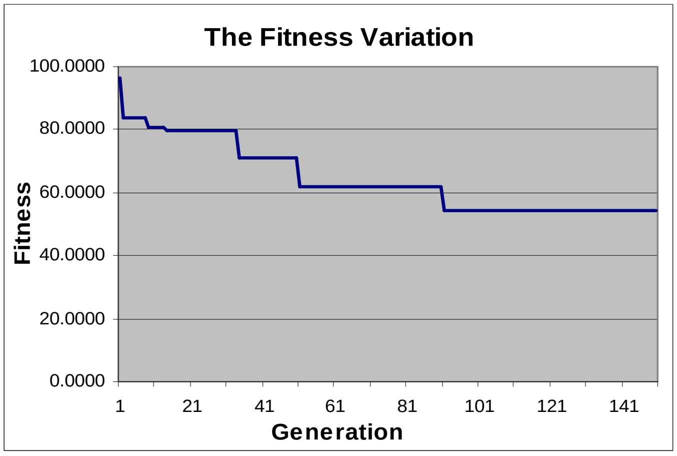

Figure 5 The fitness variation.

The result of optimization process using the GAs is shown in Table 2. It can be seen that application of GAs results in the system parameter values, which give the most optimum recognition. The NFS parameter values are applied for initializing the parameter of the system during training and testing. The best result is presented in Table 3. The comparison of NFS examination result with GAs (GNFS) and without GAs is depicted in Table 4.

Table 2 The Value of NFS parameters as the result of the GAs optimisation.

α	β	δ	Tf	M	Fitness (%)	Error Rate	Reduction Rate (%)
3.0355	0.2064	0.2011	0.2824	119	54.09	0.09	45.91

Table 3 The NFS training and testing result using optimization parameters values.

TRAINING					TESTING			
# input pattern	# neuron on 3rd and 4th layer	Reduction Rate (%)	Recog-nition Error Rate (%)	Duration	# input pattern	Recog-nition Rate (%)	Recog-nition Error Rate (%)	Duration
220	119	45.91	6	00:03.3	2200	70.36	29.64	01:03.6
550	297	46.00	9	00:19.2	2200	84.82	15.18	02:30.2
1100	673	38.82	9	01:21.5	2200	96.95	3.05	04:47.1
1650	1087	34.12	9	03:15.7	2200	99.45	0.55	08:37.1
2200	1491	32.23	10	06:07.5	2200	99.68	0.32	17:30.6

Table 4 Testing result of two models NFS classification.

Testing NFS without optimization				Testing NFS with optimization			
# Neuron on 3rd and 4th layer	Recog-nition Rate (%)	Recog-nition Error Rate (%)	Duration	# Neuron on 3rd and 4th layer	Recog-nition Rate (%)	Recog-nition Error Rate (%)	Duration
127	66.23	33.77	01:37.8	119	70.36	29.64	01:03.6
315	81.23	18.77	08:46.4	297	84.82	15.18	02:30.2
705	95.68	4.32	13:53.2	673	96.95	3.05	04:47.1
1130	98.91	1.09	19:56.6	1087	99.45	0.55	08:37.1
1563	99.45	0.55	20:36.5	1491	99.68	0.32	17:30.6

The result in Table 4 can be explained as follows:

- Application of GAs has shown that the recognition rate increases and the recognition fail rate decreases.

- The classification time increases because the number of the neuron in 3rd and 4th layer decreases, allowing the system to find the similar pattern in a faster way.

The simulation of NFS using the GBell MF as output function was compared with the Gaussian MF. The results are shown in Figures 6 and 7, GBell MF provides better performance than that of Gaussian MF.

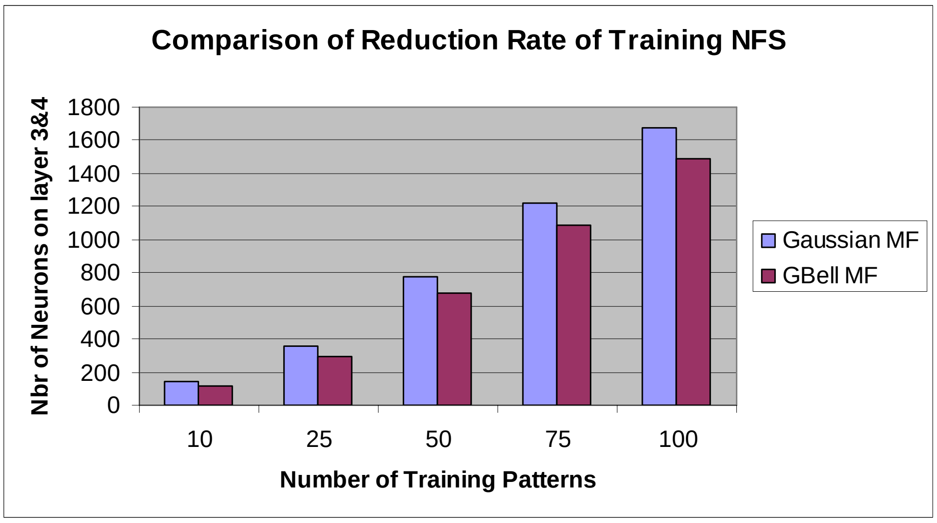

Figure 6 The Reduction Rate of the NFS using GBell MF and Gaussian MF.

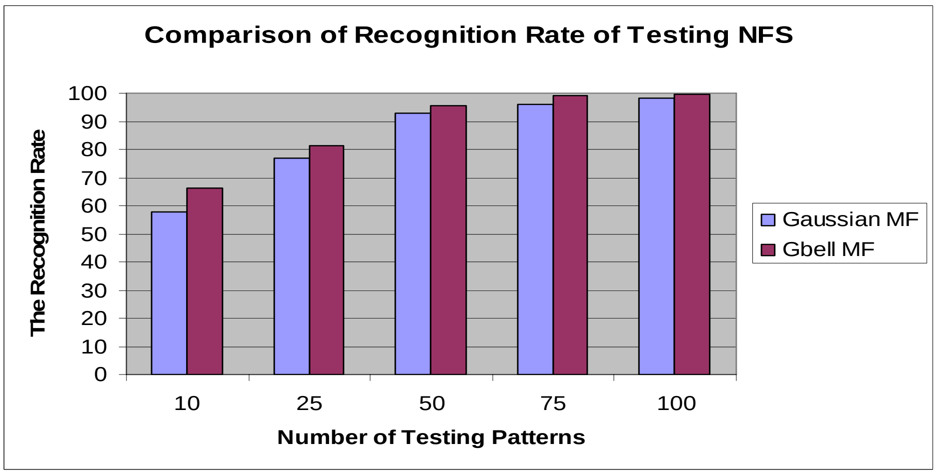

Figure 7 The Recognition Rate of the NFS using GBell MF and Gaussian MF.

Comparison of Recognition Rate of NFS and of GNFS is presented in Figure 8.

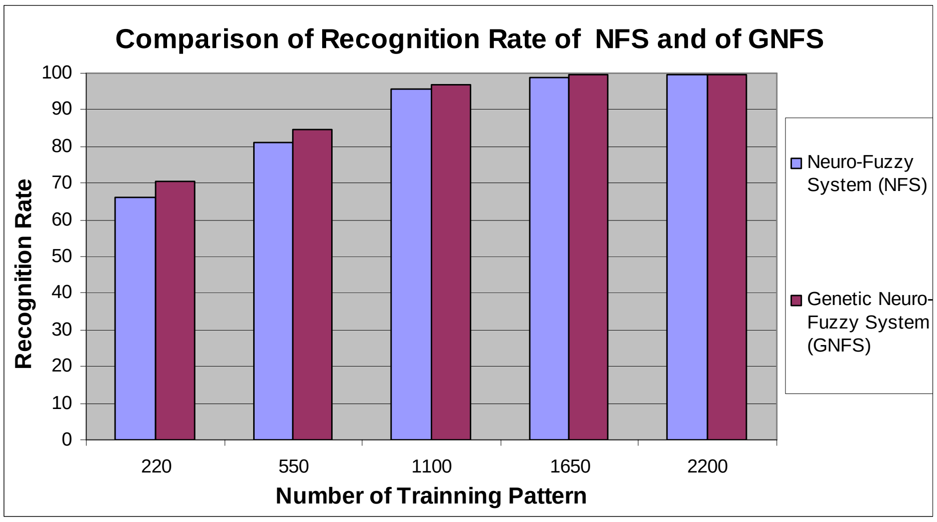

Figure 8 Comparison of Recognition rate of NFS and of GNFS.

6 Conclusion

The works showed that the NFS with GAs (GNFS) has assisted in automating the optimized setting of the parameter values, hence increased the accuracy of parameter values, accelerated the speed of training/testing, and improved the recognition rate. Our experience suggests that the GNFS provides the recognition rate that is better than that of NFS. The increase of the training pattern number will improve the test recognition rate, at the expense of slightly higher computing load. The best performance of 99.68% is achieved in an experiment with the greatest number of training pattern, i.e. 2200 patterns.

Acknowledgement

The first author would like to thank the General Director for Higher Education of the Departement of National Education of te Republik Indonesia, who has provided the BPPS scholarship.

References

[1] Vidal, E., & Castro, M.J., *Classification of Banded Chromosome using Error Correcting Grammatical Inference (ECGI) and Multilayer Perceptron (MLP*), Proc. of The 7th National Symposium on Pattern Recognition and Image Analysis, pp. 31-36, 1997.

[2] Lundsteen C., Phillip J. & Granum E., *Quantitative analysis of 6985 digitized trypsin G-banded human metaphase chromosomes*, Clinical Genetics **18**, pp. 355-370, 1986.

[3] Granum E. & Thomason M.G., *Automatically Inferred Markov Network Models for Classification of Chromosomal Band Pattern Structures*, Cytometry 11, pp 26-39, 1990.

[4] Errington P.A. & Graham J., *Application of artificial neural networks to chromosome classification*, Cytometry, pp. 627−639, 1993.

[5] Juan, A. & Vidal, E., *On the Use of Normalized Edit Distances and an Efficient k-NN Search Technique (k-AESA) for fast and Accurate String Classification*, ICPR 2, pp 680-683, 2000.

[6] Conroy, J.M., Tamara G..K., O'Leary D.P., & O'Leary, T.J., *Chromosome Identification Using Hidden Markov Models: Comparison with Neural Networks, Singular Value Decomposition, Principal Components Analysis, and Fisher Discriminant Analysis*, Center for Computing Sciences, Institute for Defense Analyses, Bowie, Maryland, USA, 2000.

[7] Schwartzkopf, W.C., *Maximum Likelihood Techniques for Joint Segmentation-Classification of Multi-spectral Chromosome Images*, Doctoral Dissertation, The Faculty of the Graduate School of The University of Texas at Austin, 2002.

[8] Sexton, R.S., Dorsey, R.E., Johnson, J.D., *Optimization of Neural Networks: A Comparative Analysis of the Genetic Algorithm and Simulated Annealing*, Department of Management Ball State University Muncie, Indiana USA, 1996.

[9] Sarosa, M., Ahmad A.S., Riyanto, B. & Noer, A.S. *Jaringan Neuro-Fuzzy (JNF) untuk mengklasifikasi Citra Kromosom Manusia*, Procceding Seminar on Intelligent Technology and Its Application'06 **2**, pp. I28-I33, 2006.

[10] Abraham, *A., EvoNF: A Framework for Optimization of Fuzzy Inference Systems using Neural Network Leraning and Evolutionary Computation*, School of Business Systems, Monash University, Clayton, Victoria Australia, 2003.

[11] Kwan H.K. & Cai Y., *A Fuzzy Neural Network and its Application to Pattern Recognition*, IEEE Trans. On Fuzzy Systems **2**, pp. 185-193, 1994.

[12] Jang J.S.R., Sun C.T. & Mizutani E., *Neuro-Fuzzy and Soft Computing*, Prentice Hall, 1997.

[13] Riyanto, B. & Sarosa, M., *Pengenalan Tulisan Tangan Angka Menggunakan Jaringan Neuro-Fuzzy*, Majalah Ilmiah Teknik Elektro **9**, pp. 1-10, 2003.

[14] Holland, J.H., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[15] Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[16] Attia, A.F. & Horacek, P., *Optimization of Neuro-Fuzzy Modeling Using Genetic Algorithm*, Proc. of XXVI. ASR' 2001 Seminar, Instruments And Control, Ostrava, Czech Republic, pp.5-15, 2001.

[17] Sexton, R.S., Dorsey, R.E., Johnson, J.D., *Optimization of Neural Networks: A Comparative Analysis of the Genetic Algorithm and Simulated Annealing*, Department of Management Ball State University Muncie, Indiana USA, 1996.

[18] Cordon, O., Gomide, F., Herrera, F., Hoffmann, F. & Magdalena, L, Ten *years of Genetic Fuzzy Systems: Current Framework and New Trends*, Fuzzy Sets and Systems **141**, pp. 5-31, 2004.

[19] Topchy, A.P., Miagkikh, V.V., Kononenko, R.N. & Melikhov, A.N., *Adaptive Genetic Search for Optimization of Fuzzy and Neuro Fuzzy Systems*, Scientific Research Institute for Multiprocessor Computer Systems, Chekhov, Taganrog, Russia, 1996.