Exploiting Diversity of Neural Ensembles with Speciated Evolution

Seung-Ik Lee, Joon-Hyun Ahn, and Sung-Bae Cho

Dept. of Computer Science, Yonsei University
Seoul 120-749, South Korea
cypher@candy.yonsei.ac.kr, jhahn@candy.yonsei.ac.kr, sbcho@csai.yonsei.ac.kr.

Abstract

In this paper, we evolve artificial neural networks (ANNs) with speciation and combine them with several methods. In general, an evolving system produces one optimal solution for a given problem. However, we argue that many other solutions exist in the final population, which can improve the overall performance. We propose a new method of evolving multiple speciated neural networks by fitness sharing that helps to optimize multi-objective functions with genetic algorithms, and several combination methods to construct ensembles of ANNs. Experiments with the UCI benchmark datasets show that the proposed methods can produce more speciated ANNs and, thus, improve the performance by combining representative individuals with combination methods.

1 Introduction

Combining multiple evolved ANNs has been actively researched recently [11, 14, 17]. Generally, multiple ANNs in the last generation are combined to construct an ensemble that has better generalization performance provided that the components, i.e., ANNs, complement each other in generalization. In terms of ANN, it means that an ANN produces does not have no or little common errors with other ANNs such that these errors can be corrected by other ANNs [11, 15, 9]. However, the ANNs in the last generation of evolution tend to be similar to each other because a high fitness individual prevails in the population after certain generations. To maximize the effect of combining multiple ANNs, therefore, a method for large diversity of neural networks in evolution should be contrived. For example, Liu and Yao have reported a series of results with negative correlation learning [9].

In this paper we propose another method of evolving ensemble neural networks with a speciation technique called fitness sharing [1, 4, 5] to generate a population of ANNs that are accurate and diverse. Speciation in genetic algorithm creates different species, each embodying a sub-solution, which means to create not only the best one but also diverse solutions [1, 5]. Especially, we adopt an information theoretic measure for the comparison of similarity of ANNs, which is used to speciate ANNs.

Assuming that the diversity of evolution can be maintained by fitness sharing, this does not guarantee that any combinations of evolved ANNs can show best performance on given problems. How to combine them is another problem. There are several methods for combining the outputs of ANNs such as average, weighted average, Dempster-Shafer methods, combining using rank-based information, voting, supra Bayesian approach, stacked generalization, etc. However, the methods are usually heavily dependent on the training data and need much knowledge on the problem.

In this paper, we will apply several combination methods for the construction of ensembles of multiple ANNs evolved with speciation and compare test results with other works.

2 Evolution of Artificial Neural Networks

Each ANN in the ensemble is generated with random initial weights and full-connection, trained partially to help the evolution search the optimal ANN architecture, and tested with validation data to compute the fitness. The fitness of an ANN is the recognition rate of validation data and is modified using speciation technique. All fitnesses calculated, selection is conducted. Genetic operators, crossover and mutation, are applied to the selected individuals to produce next generation. The process is repeated until stop criterion is satisfied. The ANNs in the last generation are trained fully.

Matrix encoding scheme is used to encode an ANN. When N is the total number of nodes of an ANN, a $N \times N$ matrix represents connection links of an ANN and their corresponding weights. Figure 1 shows the encoding example and its decoded ANN. In the matrix, upper right triangle has connection link information, where 1 means that a connec-

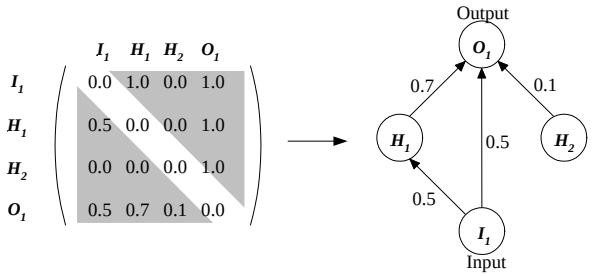


Figure 1: An example of encoding an ANN.

tion link exists and 0 in case of no connection link. Lower left triangle describes the weight value. The matrix in the figure is decoded as a ANN with one input node, two hidden nodes, and one output node as in the right side of figure 1.

For crossover, two hidden nodes from two distinct, randomly selected ANNs are chosen. These two nodes should be in the same entry of each ANN matrix to exchange the architectures. With these nodes, the two ANNs exchange the connection links and corresponding weights information of the nodes and the hidden nodes after that. Figure 2 shows an example of crossover, where H_2 node is selected as crossover points. Mutation operator changes a connection link and a corresponding weight of a randomly selected ANN from the population. Mutation operator either adds a new connection or removes an existing connection.

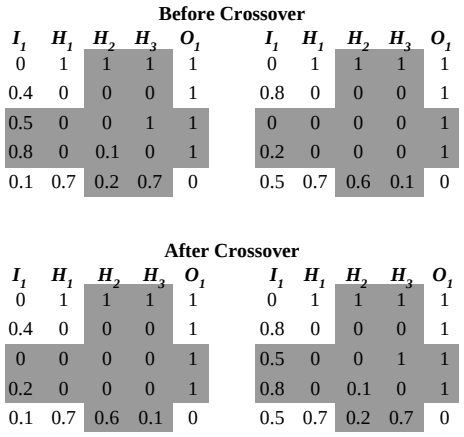


Figure 2: An example of crossover operation.

3 Speciation of Evolutionary ANNs

3.1 Fitness Sharing

Speciation can be implemented by many ways. In this work, we use fitness sharing technique [1, 4, 5]. Fitness sharing helps genetic algorithm search various space and generate ANNs that are more diverse. Let f_i be the fitness of an individual i, $sh(d_{ij})$ be sharing function, and P be the population size, then the shared fitness fs_i is computed as :

$$fs_i = \frac{f_i}{\sum_{j=1}^{P} sh(d_{ij})} \tag{1}$$

The sharing function $sh(d_{ij})$ is computed using the distance value d_{ij} that means the difference between individual i and j as follows :

$$sh(d_{ij}) = \begin{cases} 1 - \frac{d_{ij}}{\sigma_s} & \text{for } 0 \le d_{ij} < \sigma_s \\ 0 & \text{for } d_{ij} \ge \sigma_s \end{cases} \tag{2}$$

where σ_s describes the sharing radius. If the difference is larger than σ_s, they do not share the fitness.

3.2 Similarity Measures for Artificial Neural Networks

To calculate the difference between two ANNs, the average of the outputs of each ANN and modified Kullback-Leibler entropy are used. The average of the outputs of an ANN is as follows:

$$out_{avg} = (\sum_{i=1}^{N} out_i)/N \tag{3}$$

where out_i is the output of the ith input data of N data. The difference between two ANNs is the Euclidean distance of their average outputs.

Another similarity measure is modified Kullback-Leibler entropy. As the outputs of ANNs are not just likelihood or binary logical values near zero or one, but estimates of Bayesian a posteriori probabilities of a classifier, we can measure the difference between two ANNs with modified Kullback-Leilber entropy [3, 8], which is called relative entropy or cross-entropy. Let p and q be two distributions, then the distance between them is defined as follows:

$$D(p,q) = \sum_{i=1}^{m} p_i \log(\frac{p_i}{q_i}) \tag{4}$$

However, the entropy is not a true distance due to the fact that is not symmetric, i.e., $D(p,q) \neq D(q,p)$. To remedy this problem, we can define symmetric relative entropy as follows:

$$D(p,q) = \frac{1}{2} \sum_{i=1}^{m} \left(p_i \log(\frac{p_i}{q_i}) + q_i \log(\frac{q_i}{p_i}) \right) \tag{5}$$

Let p and q be output probability distributions of two ANNs which consist of m output nodes and are trained with n data. Then, the similarity of the two ANNs can be calculated as follows:

$$D(p,q) = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + q_{ij} \log(\frac{q_{ij}}{p_{ij}}) \right) \tag{6}$$

where p_{ij} is the ith output value of the ANN with respect to the jth training data. Two ANNs are more similar as the symmetric relative entropy gets smaller.

The last similarity measure we use is Pearson correlation. It can show the relation between two variables. The similarity between ANN a and u can be calculated as in equation (7).

$$w_{au} = \frac{\sum_{i=1}^{m}(r_{ai} - \overline{r}_a) \times (r_{ui} - \overline{r}_u)}{\sqrt{\sum_{i=1}^{m}(r_{ai} - \overline{r}_a)^2 \times \sum_{i=1}^{m}(r_{ui} - \overline{r}_u)^2}} \quad (7)$$

where $\overline{r}_{ai}$ is the ith output of ANN a and $\overline{r}_a$ is the average of the outputs of ANN a.

4 Combining Multiple ANNs

Combining multiple ANNs can be viewed as determining $F(e(x))$ using the output of $e(x)$ of K ANNs when the ANNs are applied to classification of pattern x into $M(A = \{1, \ldots, M\})$ classes. For convenience of discussion, we define $e_k(x)$ as the output of kth ANN.

Voting. The result supported by majority of multiple ANNs is the output of the ensemble as in equation (8).

$$F(e(x)) = j \quad \text{if } S(x \in C_j) = \max(S(x \in C_i)) \quad (8)$$

where $S(x \in C_i) = \sum_{k=1}^{K} G_k(x \in C_i), i \in A$ and $G_k(x \in C_i)$ is 1 if $e(x) = i$ and $i \in A$ else 0. Although voting has merits in that it does not require extra computations or memory space, it may aggravate the ensemble's performance if it has poor ANNs because all the ANNs have same significance regardless of the ANNs' performances.

Bayesian Method. Bayesian method takes each ANNs significance into accounts by allowing the error possibility of each ANN to affect the ensemble's results [2]. The error possibility is represented by $M \times M$ matrix, where a matrix element $n_j^i(k)$ is the number of data which belong to class C_i while the kth ANN outputs j. This error possibility is used to calculate the possibility of kth ANN classifies input x as class j whilst x belongs to class i, $P(x \in C_i | e_k(x) = j)$. The reliability function, $BEL(i)$, is represented as in equation (9).

$$BEL(i) = \eta \prod_{k=1}^{K} P(x \in C_i | e_k(x) = j_k), j=1,\ldots, \text{M} \quad (9)$$

Finally, the combination function, F, is defined as in equation (10).

$$F(e(x)) = \begin{cases} j & \text{if } BEL(j) = \max_{i \in A} BEL(i) \geq \alpha (0 < \alpha \leq 1) \\ reject & \text{otherwise} \end{cases} \quad (10)$$

where α is a threshold.

BKS Method. BKS [7] is a set of cells, where M^k cells are required to store the necessary information of K classifiers with M classes. The combination function of BKS is defined as in equation (11).

$$F(e(x)) = \begin{cases} R_{e_1(x)\ldots e_K(x)} \\ \quad \text{if } T_{e_1(x)\ldots e_K(x)} > 0 \text{ and } \frac{n_{e_1(x)\ldots e_K(x)}(R_{e_1(x)\ldots e_K(x)})}{T_{e_1(x)\ldots e_K(x)} \geq \lambda} \\ M+1 \quad \text{otherwise} \end{cases} \quad (11)$$

where $n_{e_1(x)\ldots e_K(x)}(m)$ is the number of data belonging to class m, $T_{e_1(x)\ldots e_K(x)}$ is the total number of data belonging to $BKS(e_1(x), \ldots, e_K(x))$, and $R_{e_1(x)\ldots e_K(x)}$ is the representative class of $BKS(e_1(x) \ldots e_K(x))$ with $BKS(e_1(x) \ldots e_K(x))$ being a cell with index $(e_1(x) \ldots e_K(x))$.

Borda Function. Let r_k^i be the ranking of ith class of kth classifier, then Borda score is calculated by summing $M - r_k^i$. The output of an ensemble is determined by the following equations.

$$F(e(x)) = \max_{i \in A} \sum_{k=1}^{K} (M - r_k^i(x)) \quad (12)$$

Condorect Function. With Condorect function, the output of an ensemble is the class with the highest Condorect value. The Condorect value of a class is the minimum among the numbers of classifiers that rank the class higher than other classes. Mathematically, the output of an ensemble is determined by equation (13).

$$\begin{aligned} F(e(x)) &= \max_{i \in A}(Con(i)) \\ \text{with } Con(i) &= \min_{j \in M-i} \#(K : r_k^i > r_k^j) \end{aligned} \quad (13)$$

where $\#(K : r_k^i > r_k^j)$ is the number of ANNs that rank class i higher than j.

Average. The output of an ensemble in this method is determined as the class with the highest average value of the summation of output m_k^i of each class i as in equation (14).

$$F(e(x)) = \max_{j \in A} S_j, \; S_i = (\sum_{k=1}^{K} m_k^i)/K \quad (14)$$

Weighted Average. Weighted average [11] multiplies weight w to the output of each ANN when averaging the

outputs. Let E_i be the error rate of ith ANN, then the weight w_i is computed as follows:

$$w_i = \frac{1 - E_i}{\sum_k (1 - E_k)} \quad (15)$$

Gating. Gating is a method for choosing the fittest ANN by utilizing the information on learning data. Four steps are required to do this.

- Step 1 : In learning stage, all ANNs generate data list for which they have generated correct outputs.
- Step 2 : In test stage, search learning data most similar to input data.
- Step 3 : Choose an ANN which have recognized the searched learning data correctly.
- Step 4 : The chosen ANN classifies the test data.

Ideal Method. This chooses the output of an ANN, if any, that classifies the given test data correctly. Therefore, as long as ANNs that produce a correct answer exist, the whole ensemble produces a correct answer. This method is for the comparisons with other combination methods.

5 Experimental Results

5.1 Experimental Setup

To show the effectiveness of the proposed method, some experiments are conducted for benchmark problems, Australian credit approval, breast cancer, and diabetes from UCI machine learning dataset.

Breast cancer data set is a 2-class problem with 699 examples. Each data has 9 attributes and 1 class attribute. We have used training, validation, and test data sets respectively with 349, 175, and 175 examples. Australian credit approval data set is a 2-class problem with 690 examples. Each data has 14 attributes. We have used training, validation, and test data sets respectively with 346, 172, and 172 examples. Diabetes data set has 768 examples. Five hundred examples out of 768 are one class and the others are two classes. Each data has nine attributes and one class attribute. We have used training, validation, and test data sets respectively with 384, 192, and 192 examples.

Population size is 20 and the maximum generation number is 200. Crossover rate is 0.3 and mutation rate is 0.1. Each ANN is a feed-forward ANN with five hidden node trained by back-propagation with learning rate 0.1. In partial training, training data is presented 200 times and 1000 times in full training.

5.2 Analysis of Speciation

Table 1 shows the recognition rates in relation to speciation methods. Average recognition rates of both speciated systems are lower than that of 'no speciation'. This means each individual of 'no speciation' has better recognition ability than that of speciated EANNs.

Table 1: Recognition rates. Subscript A=Average Output; E=Entropy; N=No Speciation.

	Avg	StdDev	Max	Min
Train_A	0.9469	0.0095	0.9628	0.9226
Train_E	0.9370	0.0207	0.9656	0.8883
Train_N	0.9509	0.0124	0.9685	0.9112
Varify_A	08951	0.0177	0.9143	0.8457
Varify_E	0.8683	0.0340	0.9200	0.7886
Varify_N	0.9009	0.0270	0.9314	0.8171
Test_A	0.9494	0.0176	0.9714	0.9086
Test_E	0.9349	0.0271	0.9714	0.8571
Test_N	0.9554	0.0179	0.9771	0.9143

We have used single linkage cluster analysis to analyze the speciation of ANNs and select representative ANNs from each speciation [6]. By combining the results of the representatives of each species, the speciated EANNs have better performance than EANNs with no speciation in the case of average and weighted average as in table 2. Though the EANNs with no speciation have generated better individual ANNs, they have little performance increase by combination. It can be inferred that the generated ANNs in this system are all similar so that combining them have not resulted in much increase of performance. Although the speciated EANNs have generated individual ANNs that have worse recognition rate, their combinations have better performance than those of 'no speciation', indicating that the individuals of speciation complement each other and the speciated EANNs have generated diverse ANNs with speciation.

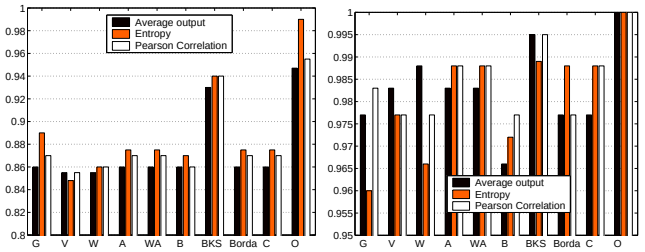
Table 2: The comparison of the proposed method and multiple EANNs with no speciation.

	Vote	Avg	Wavg	Gating	Opt
AvgOut	.9634	.9806	.9806	.9577	.9954
Entropy	.9588	.9794	.9783	.9520	.9989
No spec.	.9668	.9777	.9777	.9577	.9966

5.3 Analysis of the Combination Methods

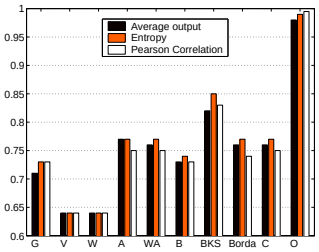
Figure 3 (a) shows the combination results of the speciated ANNs for the data sets. In terms of similarity measures for ANNs, Kullback-Leibler entropy shows good recognition results than other speciation methods do. In terms of combination methods, BKS shows best result except for ideal

method. Figure 3 (b) shows the results of breast cancer data. 'Voting,' 'Winner,' and 'BKS' show good recognition results with 'Average output' and 'Average,' 'Weighted Average,' and 'Borda' show good recognition results with 'Entropy'. 'Pearson Correlation' shows good results when 'Gating,' 'Average,' 'Weighted Average,' 'Bayesian,' and 'BKS' are used. The BKS combination method outperforms other combination methods, which is also the same with Australian credit approval data. Figure 3 (c) shows the results of diabetes data. In terms of similarity measures, entropy method shows good overall performance than others do and BKS shows best performance. In summary, entropy



(a) Australian credit approval.

(b) Breast cancer.



(c) Diabetes.

Figure 3: Test on UCI benchmark data. Here, G=Gating, V=Voting, W=Winner, WA=Weighted Average, B=Bayesian, C=Condorect, and O=Ideal.

method shows good performance in terms of similarity measures, and BKS ranks the first in all the three-benchmark data in terms of combination methods.

To estimates the performance of this work, comparisons with other works have been done. For this, results of the BKS combination method have been used. Table 3 shows the results of the BKS combination method and table 4 shows comparisons of the BKS combination with the works on Australian credit approval data set by Michie [10]. The BKS combination method shows low error rate than other works do. Especially, speciation with average output shows best performance. Table 5 shows comparisons with other representative works, FNNCA [13], HDANNS [12], and

Table 3: BKS combination method : Subscript A=Australian credit approval; B=Breast Cancer; D=Diabetes.

	Avg	Entropy	Pearson
Average_A	0.9122	0.9041	0.9076
Average_B	0.98745	0.98916	0.98745
Average_D	0.7823	0.7979	0.7948
StdDev_A	0.0108	0.0151	0.0151
StdDev_B	0.006489	0.007836	0.005906
StdDev_D	0.0207	0.0320	0.0249
Max_A	0.9302	0.9360	0.9360
Max_B	0.9943	1.0000	0.9943
Max_D	0.8125	0.8438	0.8281
Min_A	0.8953	0.8779	0.8895
Min_B	0.9771	0.9714	0.9771
Min_D	0.7448	0.7396	0.7396

Table 4: Comparisons with other works : Audstralian credit approval.

	Avg	Pearson	Entropy	EPNet
Error	0.089	0.092	0.096	0.115
	Cal5	ITrule	DIPOL92	Discrim
Error	0.131	0.137	0.141	0.141
	Logdisc	CART	RBF	CASTLE
ERror	0.141	0.145	0.145	0.148
	NaiveBay	IndCART	BP	×
Error	0.151	0.152	0.154	×

EPNet [16]. FNNCA is an ANN with constructive algorithm and HDANNS is an optimized ANN constructed by hand with trial and error. EPNet is an evolutionarily constructed ANN by Yao and Liu. The comparisons show that the BKS combination of speciated ANNs outperforms FNNCA and EPNet. Although HDANNS shows better performance than BKS, BKS can be considered to show good performance considering the cost-high construction process of HDANNS. Table 6 shows test results of BKS and comparisons with other works, respectively. The BKS combination with speciation shows best results than others do.

Table 5: Comparisons with other works : Breast cancer.

	Entropy	Avg	Pearson
Error	0.0118	0.0125	0.0125
	HDANNS	EPNet	FNNCA
Error	0.01149	0.01376	0.0145

Table 6: Comparisons with other works : Diabetes.

	Entropy	Pearson	Avg	Logdisc
Error	0.202	0.205	0.218	0.223
	EPNet	DIPOL92	Discrim	SMART
Error	0.224	0.224	0.225	0.232
	RBF	ITrule	BP	Cal5
Error	0.243	0.245	0.248	0.250
	CART	CASTLE	Quadisc	×
Error	0.255	0.258	0.262	×

6 Concluding Remarks

In this paper, we have applied several similarity measures for ANNs to speciate them and constructed ensembles of ANNS with several combination methods. We have used 'average output,' 'Kullback-Leibler,' and 'Pearson correlation' methods for measuring the similarity between two ANNs.

We have shown that BKS outperforms other combination methods on the three UCI machine learning benchmark data, Australian credit approval, breast cancer, and diabetes. Nevertheless, more works on combination methods should be done to decrease the differences with 'Ideal' method. We have also shown that speciation shows better performance than no speciation does when combined with ensemble combination methods. We infer from this fact that genetic drift keeps ANNs from being diverse and, thus, the combination of these ANNS shows little performance increase.

As a further research, works should be done on more similarity measures and determination of the sharing distance.

References

[1] T. Bäck, D.B. Fogel and Z. Michalewicz, *Evolutionary Computation 2 : Advanced Algorithms and Operators*, IOP, 2000.

[2] J.C. Bioch, O.V.D. Meer, R. Potharst, "Classification using Bayesian neural nets," *IEEE International Conference on Neural Networks*, vol. 3, pp. 1488-1493, 1996.

[3] T. Cover and J. Thomas, *Elements of Information Theory*, Wiley Series in Communications, New York, 1991.

[4] D.E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," *Proc. of the Second Int. Conf. on Genetic Algorithms*, pp. 41-49, 1987.

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

[6] A.D. Gordon, *Classification : Methods for the Exploratory Analysis of Multivariate Data*, Chapman and Hall, 1981.

[7] A. Khotanzad, and C. Chung, "Hand written digit recognition using BKS combination of neural network classifiers," *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation*, pp.94-99, 1994.

[8] S. Kullback and R.A. Leibler, "On Information and Sufficiency," Ann. Math. Stat., 22, pp. 79-86, 1951.

[9] Y. Liu, X. Yao and T. Higuchi, "Evolutionary ensembles with negative correlation learning," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 380-387, November 2000.

[10] D. Michie, D.J. Spiegelhalter, and C.C. Taylor, *Machine Learning, Neural and Statistical Classification*, Ellis Horwood, 1994.

[11] D.W. Opitz and J.W. Shavlik, "Actively searching for an effective neural network ensemble," *Connection Science*, vol. 8, No. 3 & 4, pp. 337-353, 1996.

[12] L. Prechelt, "Proben1 - a set of neural network benchmark problems and benchmarking rules," *Tech. Rep. 21/94*, Fakultatfur Informatik, Universitat Karlsruhe, 76128, Germany, 1994.

[13] R. Setino and L.C.K. Hui, "Use of a quasi-newton method in a feed forward neural network construction algorithm," *IEEE Trans. on Neural Networks*, vol. 6, no. 1, pp. 273-277, 1995.

[14] A.J.C. Sharkey, "On combining artificial neural nets," *Connection Science*, vol. 8, pp.299-313, 1996.

[15] A.J.C. Sharkey and N.E. Sharkey, "Combining diverse neural nets," *The Knowledge Engineering Review*, vol. 12, no. 3, 231-247, 1997.

[16] X. Yao and Y. Liu, "A new evolutionary system for evolving artificial neural networks," *IEEE Trans. Neural Networks*, vol 8, pp.694-713, Anchorage, USA, 4-9 May 1998.

[17] X. Yao and Y. Liu, "Making use of population information in evolutionary artificial neural networks," *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, vol. 28, no. 3, pp. 417-425, June 1998.