Time Series Prediction Using Committee Machines of Evolutionary Neural Trees

Byoung-Tak Zhang
Artificial Intelligence Lab (SCAI)
Dept. of Computer Engineering
Seoul National University
Seoul 151-742, Korea
URL: http://scai.snu.ac.kr/~btzhang

Je-Gun Joung
Artificial Intelligence Lab (SCAI)
Dept. of Computer Engineering
Seoul National University
Seoul 151-742, Korea
URL: http://scai.snu.ac.kr/~jgjoung

Abstract- Evolutionary neural trees (ENTs) are tree-structured neural networks constructed by evolutionary algorithms. We use ENTs to build predictive models of time series data. Time series data are typically characterized by dynamics of the underlying process and thus the robustness of predictions is crucial. In this paper, we describe a method for making more robust predictions by building committees of ENTs, i.e. CENTs. The method extends the concept of mixing genetic programming (MGP) which makes use of the fact that evolutionary computation produces multiple models as output instead of just one best. Experiments have been performed on the laser time series in which the CENTs outperformed the single best ENTs. We also discuss a theoretical foundation of CENTs using the Bayesian framework for evolutionary computation.

1 Introduction

Time series data are typically characterized by dynamics of the underlying process and thus the robustness of predictions is crucial. One method of increasing the robustness of predictive models is to use an ensemble or committee of models.

Committee machines can improve predictive accuracy by reducing variance due to the averaging over many models. Averaging is effective since the construction of models is an optimization problem with many local minima. All global optimization methods in the face of many local minima yield "optimal" parameters which differ greatly from one run of the algorithm to the next, i.e., which show a great deal of randomness stemming from different initial points. This randomness tends to differentiate the errors of the models, so that the models will be making errors on different subsets of the input space. As each network makes generalization errors on different subsets of the input space, the collective decision produced by the committee is less likely to be in error than the decision made by any of the individual models.

From the evolutionary computation point of view, committee machines are natural; every evolutionary algorithm produces a number of individuals as their result. In fact, Yao and Liu [9] and Zhang and Joung [11] have independently developed methods for combining multiple individuals evolved by evolutionary algorithms. The former (EPNet) used evolutionary programming to evolve multilayer perceptrons while the latter (MGP) is based on genetic programming to construct evolutionary neural trees (ENTs). Both have shown that combining multiple models improves generalization performance of models generated by evolutionary algorithms.

In this paper, we extend the concept of mixing genetic programming (MGP) to further improve its performance by using a new training method for committee members. As before, we use evolutionary neural trees (ENTs) to build predictive models of time series data. In previous work [11] we used weighted majority algorithm to learn the combination coefficients of ENTs. The committee of evolutionary neural trees or CENTs is now trained by taking into account the correlation between the errors made by the members. The rationale behind this approach is the theoretical observation that minimizing the correlation between the committee members maximizes the ensemble effect [8] (more details in Section 3 below).

The paper is organized as follows. In Section 2 we describe the structure of neural trees for modeling time series data and the evolutionary algorithm for evolving them. Section 3 presents the method for building a committee from the individual neural trees evolved. Section 4 reports experimental results for laser-generated time series data. Section 5 contains conclusions and some remarks on Bayesian aspects of the committees of ENTs.

2 Evolutionary Neural Trees

2.1 Neural Trees

Let $\mathcal{NT}(d, b)$ denote the set of all possible trees of maximum depth d and maixmum b branches for each node. The nonterminal nodes represent neural units and the neuron type is an element of the basis function set $\mathcal{F} = \{\text{neuron types}\}$. Each terminal node is labelled with an element from the terminal set $\mathcal{T} = \{x_1, x_2, ..., x_n\}$, where x_i is the ith component of the external input $\mathbf{x}$. Each link (j, i) represents a directed connection from node j to node i and is associated with a value w_{ij}, called the synaptic weight. The members of $\mathcal{NT}(d, b)$ are referred to as neural trees. In case of $\mathcal{F} = \{\Sigma, \Pi\}$, the trees are specifically called sigma-pi neural trees. The root node is also called the output unit and the terminal nodes are called input units. Nodes that are not input or output units are hidden units. The layer of a node is defined as the longest path length to any terminal node of its subtree. Different neuron types are distinguished in the way of computing net inputs. For the evolution of higher-order networks, we consider two types of units. Sigma units compute the sum of weighted inputs from the lower layer:

$$net_i = \sum_j w_{ij} y_j \tag{1}$$

where y_j are the inputs to the ith neuron. Pi units compute the product of weighted inputs from the lower layer:

$$net_i = \prod_j w_{ij} y_j \tag{2}$$

where y_j are the inputs to i. The output of a neuron is computed either by the threshold response function

$$y_i = \sigma(net_i) = \begin{cases} 1 & : \quad net_i \geq 0 \\ -1 & : \quad net_i < 0 \end{cases} \tag{3}$$

or the sigmoid transfer function

$$y_i = f(net_i) = \frac{1}{1 + e^{-net_i}} \tag{4}$$

where net_i is the net input to the unit computed by Eqn. 1 or 2.

A higher-order network with m output units can be represented by m sigma-pi neural trees. That is, the genotype A_i of ith individual in our evolutionary framework consists of m neural trees.

The neural tree representation does not restrict the functionality since any feedforward network can be represented with a forest of neural trees:

$$A_i = (A_{i,1}, A_{i,2}, ..., A_{i,m})$$

where $A_{i,k} \in \mathcal{NT}(d, b)\ \forall k \in \{1, ..., m\}$.

The connections between input units to arbitrary units in the network is also possible since input units can appear more than once in the neural tree representation. The output of one unit can be used as input to more than one unit. The duplication does not necessarily mean more space requirements in trees than network representations since frequently-used fit submodules can be stored and multiply reused. This leads to the construction of modular structures and reduces memory requirements for representing the population [12].

Neural trees do not require decoding for their fitness evaluation. Training and evaluation of fitness can be performed directly on the genotype since both the genotype and phenotype are equivalent. Since subtree crossover used in genetic programming may be applied without modification to this representation, we can use genetic programming as the main evolutionary engine.

2.2 Evolving Neural Trees

For the construction of neural models, we maintain a population $\mathcal{A}$ consisting of M individuals of variable size:

$$\mathcal{A}(g) = \{A_1, A_2, ..., A_M\}. \tag{5}$$

Each individual A_i is a neural network represented as neural trees. The initial population $\mathcal{A}(0)$ is created at random. In each generation g, the fitness values $F_i(g)$ of networks are evaluated and the upper $\tau\%$ are selected to be in the mating pool $\mathcal{B}(g)$. The next generation $\mathcal{A}(g+1)$ of M individuals are then created by exchanging subtrees and thereby adapting the size and shape of the network. Mutation changes the node type and the index of incoming units. The best individual is always retained in the next generation so that the population performance does not decrease as generation goes on (elitist strategy).

Between generations the network weights are adapted by a stochastic hill-climbing search. This search method is based on the breeder genetic algorithm [6], in which the step size Δw is determined with a random value $\epsilon \in [0, 1]$:

$$\Delta w \quad = \quad R \cdot 2^{-\epsilon \cdot K}, \tag{6}$$

where R and K are constants specifying the range and slope of the exponential curve. In the experiments, the values were $R = 2$ and $K = 3$. This method proved very robust for a wide range of parameter-optimization problems. The fitness F_i of the individuals A_i is defined as

$$F_i(g) = F(D|A_i^g) = E(D|A_i^g) + \alpha(g)C(A_i^g), \tag{7}$$

where $\alpha(g)$ is the Occam factor [12] that adaptively balances the error $E(D|A_i^g)$ and complexity $C(A_i^g)$ of the neural trees. This evaluation measure prefers simple networks to complex ones and turned out to be important for achieving good generalization.

3 Building a Committee of Evolutionary Neural Trees

A committee consists of multiple neural trees evolved by the algorithm described in the previous section. The committee is trained to find an optimal combination weights to minimize the mean squared error between the desired and the committee's output with respect to the distribution of the training data.

The combination weights are determined by the generalized ensemble method (GEM) as proposed by Perrone [7]. It is a linear combination of the estimators based on the empirical MSE and defined as

$$f_{GEM}(\mathbf{x}) = \sum_{i=1}^{K} v_i f_i(\mathbf{x}), \tag{8}$$

where the v_i's satisfy the constraint that $\sum v_i = 1$. We want to choose the v_i's so as to minimize the MSE with respect to the target function $y(\mathbf{x})$. If we define the error $e_i(\mathbf{x})$ of member i as

$$e_i(\mathbf{x}) = y(\mathbf{x}) - f_i(\mathbf{x}) \tag{9}$$

and the correlation matrix as

$$C_{ij} = E[e_i(\mathbf{x})e_j(\mathbf{x})], \tag{10}$$

then we must minimize

$$MSE[\bar{f}] = \sum_{i,j} \alpha_i \alpha_j C_{ij}. \tag{11}$$

Each v_i is then given as

$$v_i = \frac{\sum_{j=1}^{K} C_{ij}^{-1}}{\sum_{k=1}^{K} \sum_{j=1}^{K} C_{kj}^{-1}}, \tag{12}$$

where C_{ij} are elements of the covariance matrix of the errors from the committee members f_i and f_j.

The theoretical background behind this approach can be explained by the bias-variance analysis [8]. Let $y(\mathbf{x})$ denote the target value for the input $\mathbf{x}$ and $f_i(\mathbf{x})$ the actual output of kth member of the committee. The inputs $\mathbf{x}$ are taken to be drawn from some distribution $P(\mathbf{x})$. A weighted ensemble average is denoted by

$$\bar{f}(\mathbf{x}) = \sum_{i=1}^{K} v_i f_i(\mathbf{x}), \tag{13}$$

which is the final output of the committee. v_i is the weight for combination. For an input $\mathbf{x}$ we define the error of the ensemble $\epsilon(\mathbf{x})$, the error of the ith member $\epsilon_i(\mathbf{x})$, and its ambiguity $a_i(\mathbf{x})$:

$$\begin{aligned}
\epsilon(\mathbf{x}) &= (y(\mathbf{x}) - \bar{f}(\mathbf{x}))^2 && (14)\\
\epsilon_i(\mathbf{x}) &= (y(\mathbf{x}) - f_i(\mathbf{x}))^2 && (15)\\
a_i(\mathbf{x}) &= (f_i(\mathbf{x}) - \bar{f}(\mathbf{x}))^2. && (16)
\end{aligned}$$

The ensemble error can be written as

$$\epsilon(\mathbf{x}) = \bar{\epsilon}(\mathbf{x}) - \bar{a}(\mathbf{x}) \tag{17}$$

```
1. Evolve a population of evolutionary neural trees.
2. Build a pool of candidate committee members.
3. Repeat the following $N_c$ times:
   3.1 Build a committee by members from the pool.
   3.2 Apply GEM to train the voting coefficients.
   3.3 Update the best committee if necessary.
4. Use the best committee for final prediction.
```

Figure 1: Building the best committee of evolutionary neural trees.

where $\bar{\epsilon}(\mathbf{x}) = \sum_i v_i \epsilon_i(\mathbf{x})$ is the average of the errors of the individual predictors and $\bar{a}(\mathbf{x}) = \sum_i v_i a_i(\mathbf{x})$ is the average of their ambiguities, which is simply the variance of the output over the ensemble.

All these formulas can be averaged over the input distribution $P(\mathbf{x})$ and we then obtain the ensemble generalization error

$$\epsilon = \bar{\epsilon} - \bar{a}, \tag{18}$$

where $\epsilon(\mathbf{x})$ averaged over $P(\mathbf{x})$ is simply denoted ϵ, and similarly for $\bar{\epsilon}$ and $\bar{a}$. The first term on the right is the weighted average of the generalization errors of the individual predictors, and the second is the weighted average of the ambiguities, or the ensemble ambiguity. An important feature of equation (18) is that it separates the generalization error into a term that depends on the generalization errors of the individual members and another term that contains all correlations between the individuals. The relation (18) shows that the more the predictors differ, the lower the error will be, provided the individual errors remain constant. This is the rationale behind the correlation-based training method for building committee machines.

The algorithm for building a committee of evolving neural trees is summarized in Figure 1. This algorithm is applied after the population of neural trees has evolved. Because the number of combinations is very large, we first select a pool of committee members from which a fixed size of committee is selected. Selection of committee members is performed probabilistically using a fitness-proportional mechanism with exponential scaling. A total number N_c committees are constructed and the best committee is then used for final prediction.

4 Experimental Results

Figure 2 shows a series of 2000 measurements of chaotic intensity fluctuations. This data was generated from far-infrared NH_3 laser in a physics laboratory [4]. This problem

Algorithm Parameters	Values Used
population size	200
max generation	50
crossover rate	0.95
mutation rate	0.1
no. of local descents	100
training set size	1000
test set size	1000
size of committee pool	10
no. of committee members	3

Table 1: Parameter values used in the experiments.

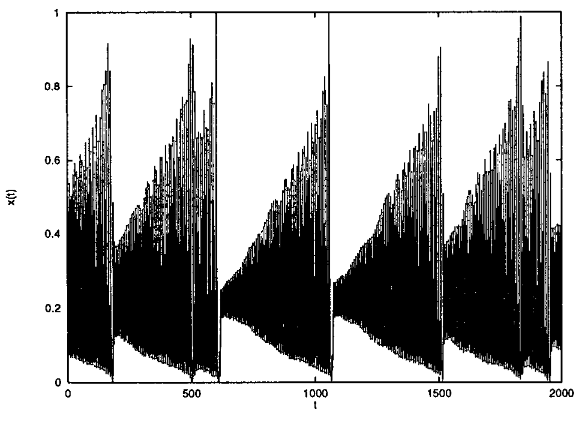

Figure 2: Laser data set: the first 1000 data points are used for training ENTs and the rest 1000 are for testing their predictive accuracy.

was used as a benchmark in the 1992 Santa Fe time series competition. We used the first 1000 data points for evolving the neural tree models and the rest 1000 data points for testing the predictive accuracy. The training data was constructed from the time series as follows: three contiguous values $x(t-3)$, $x(t-2)$, $x(t-1)$ were used as input for the tth training pattern, and the immediate next point $x(t)$ was used as the target value $y(t)$ to be predicted. The input attributes of all data sets were linearly rescaled into the interval $[0, 1]$. The output attribute has continuous values between 0 and 1.

Table 1 summarizes parameter values used in the experiments. Each run consists of 50 generations with a population size of 200. The size of committee pool was 10 and the number of members of each committee was 3.

Figure 3 compares the fitness of the best neural trees with that of the committee for the test set. This is the results averaged over 10 runs. In most generations, the committee machines outperformed the predictive accuracy of the single best. This is especially true for early generations. As generation goes on the committee effect decreases. This seems to be due to the decrease in diversity in the population. Figure 4

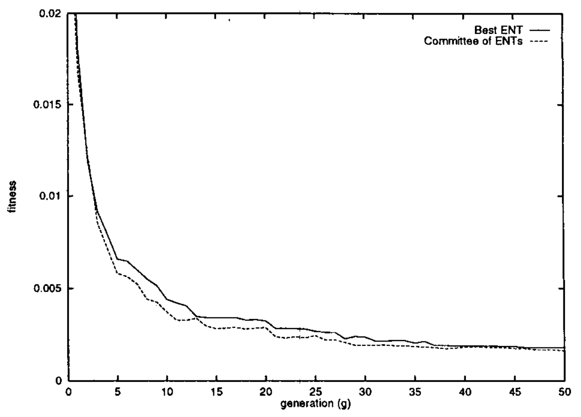

Figure 3: Comparison of fitness between the best neural tree and the committee for each generation (averaged over 10 runs).

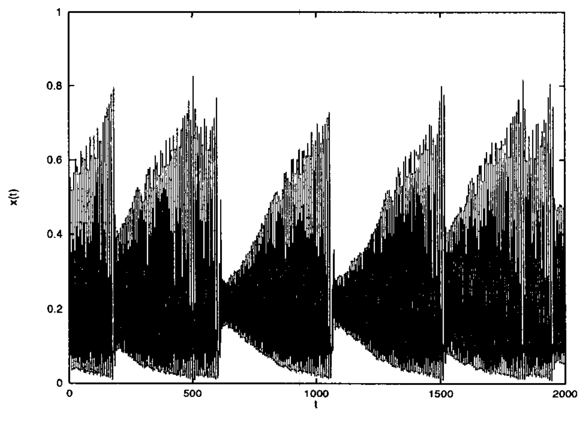

Figure 4: Performance of committees of ENTs for training ($t = 0, ..., 1000$) and for testing ($t = 1001, ..., 2000$).

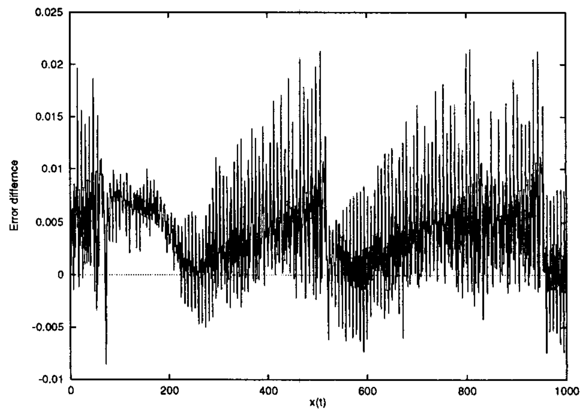

Figure 5: Difference of errors for each data point (averaged over the whole generations) between the best ENT and the committee of ENTs. Positive values mean that the committee machines outperform the single best ENT on average for the specific data point.

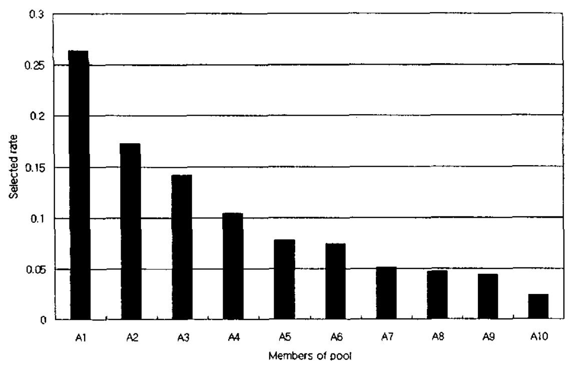

Figure 6: Histogram of the three members chosen for the best committee from the committee pool of size 10. This indicates that the best three ENTs tend to be selected as the best committee, but not always.

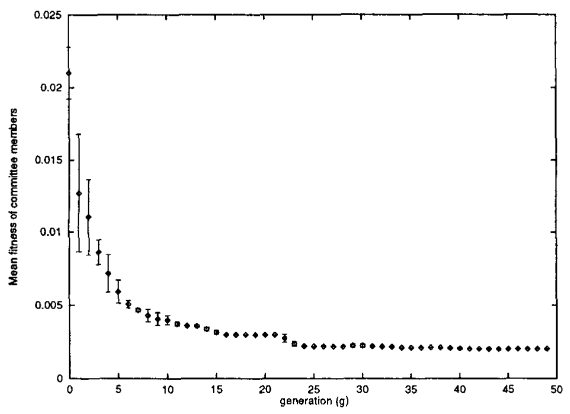

Figure 7: Standard deviation of fitness values of the committee members at each generation.

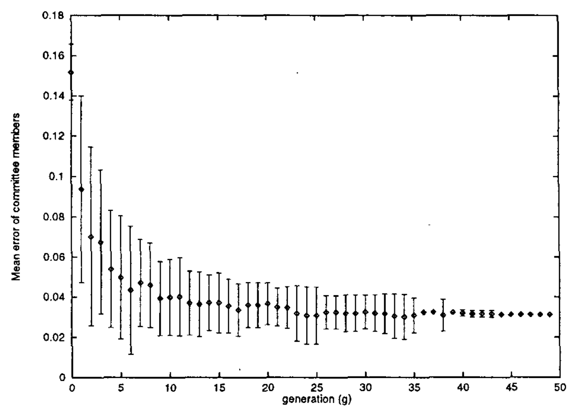

Figure 8: Standard deviation of sum of test errors of the committee members.

shows the predictive performance of a committee of ENTs.

Figure 5 shows a more detailed comparison result. To plot this, we first measured the difference in fitness values of both methods for the ith data point at gth generation:

$$d_i(g) = e_i^{best}(g) - e_i^{com}(g). \tag{19}$$

By averaging these values for the whole generations

$$d_i = \frac{1}{G}\sum_{g=1}^{G} d_i(g) \tag{20}$$

we get the plot in the figure, i.e., the average difference of errors for data point i between the best ENT and the committee of ENTs. In this plot, a positive value indicates the superiority of committee of ENTs to the single ENT in predicting the particular data point $x(t)$.

Figure 6 shows the histogram of the 3 members chosen for the best committee from the committee pool of size 10. This indicates that the best three ENTs tend to be selected as the best committee, but not always. This verifies the necessity of diversity in the committee and possibly the usefulness of reducing correlations between the committee members, as was discussed in the previous section.

Figure 7 shows the change of standard deviation of fitness values of the committee members as generation goes on. Figure 8 shows the change of standard deviation of the sum of test error values of the committee members as generation goes on. Comparing these graphs with the graph shown in Figure 3 indicates once again the usefulness of diversity and decorrelation of committee members.

5 Concluding Remarks

We have shown that committees of evolutionary neural trees (ENTs) can improve the predictive accuracy of individual ENTs applied to time series prediction. In particular, we described a new weighting scheme for committee members which take into account the correlations between committee members. This method extends the concept of the mixing genetic programming (MGP) [11] method to regression problems.

The bias-variance analysis shows that the committee effect can be maximized by combining uncorrelated predictors with large variance as long as their bias error is small. In this respect, evolutionary neural trees provide natural candidates for building committee machines since they generally have diversity in architecture and weights.

From the Bayesian evolutionary point of view [10], it is interesting to observe that the posterior probability distribution of models can be built by the committee of ENTs. By considering each neural tree in the committee as a local minima, the posterior distribution of the weights can be represented as

$$p(\mathbf{w}|D) = \sum_{i=1}^{K} p(A_i, \mathbf{w}|D) \quad (21)$$

$$= \sum_{i=1}^{K} p(\mathbf{w}|A_i, D)P(A_i|D) \quad (22)$$

where A_i denotes the neural tree evolved. This distribution is then used to predict the mean output by the committee

$$\bar{f}(\mathbf{x}) = \int f_i(\mathbf{x}, \mathbf{w})p(\mathbf{w}|D)d\mathbf{w} \quad (23)$$

$$= \sum_{i=1}^{K} P(A_i|D) \int_{R_i} f_i(\mathbf{x}, \mathbf{w})p(\mathbf{w}|A_i, D)d\mathbf{w} \quad (24)$$

$$= \sum_{i=1}^{K} P(A_i|D)\bar{f}_i(\mathbf{x}) \quad (25)$$

where R_i denotes the region of weight space surrounding the ith local minima, and $\bar{f}_i$ is the corresponding neural tree prediction averaged over this region. This indicates that the committee output is a linear combination of predictions made by each of the neural trees corresponding to distinct local minima, weighted by the posterior probability of that solution.

Acknowledgments

This research was supported in part by the Korea Ministry of Science and Technology through KISTEP under grant BR-2-1-G-06.

Bibliography

[1] Durbin, R. and Rumelhart, D. E.(1989) "Product units: a computationally powerful and biologically plausible extension to backpropagation networks," *Neural Computation*, 1: 133-142.

[2] Hansen, L. K. and Salamon, P. (1990) "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10): 993-1001.

[3] Hashem, S. (1993) "Optimal linear combinations of neural Networks," *PhD thesis, Purdue University.*

[4] Hübner, U., Weiss, C. O., Abraham, N. B. and Tang, D. (1993) "Lorenz-like chaos in NH_3-FIR laser," A. S. Weigend, N. A. Gershenfeld, (eds.) *Time Series Prediction: Forecasting the Future and Understanding the Past*, Addison-Wesley, pp. 73-104.

[5] Liu, Y. and Yao, X. (1998) "Time series prediction by using negatively correlated neural networks," *1998 Simulated Evolution and Learning Conference*, (SEAL98), Springer-Verlag.

[6] Mühlenbein, H. and Schlierkamp-Voosen, D. (1994), "The science of breeding and its application to the breeder genetic algorithm," *Evolutionary Computation*, 1(4): 335-360.

[7] Perrone M. P. and Cooper, L. N. (1994) "When networks disagree: Ensemble methods for hybrid neural networks," in *Artificial Neual Networks for Speech and Vision*, Chapman & Hall, pp. 126-142.

[8] Sollich, P. and Krogh, A. (1996) "Learning with ensembles: How over-fitting can be helpful," in *Advances in Neural Information Processing Systems*, MIT Press, pp. 190-196.

[9] Yao, X. and Liu, Y. (1998) "Making use of population informatin in evolutionary artificial nerual networks ," *IEEE Transactions on Systems, Man and Cybernetics*, 28B(3): 417-425.

[10] Zhang, B.-T. (1999) "A Bayesian framework for evolutionary computation," *Congress on Evolutionary Computation* (CEC99), Washington, D.C., IEEE Press.

[11] Zhang, B.-T., Joung, J.-G. (1997) "Enhancing robustness of genetic programming at the species level," *Genetic Programming Conference* (GP-97), Morgan Kaufmann, pp. 336-342.

[12] Zhang, B.-T., Ohm, P. and Müehlenbein, H. (1997) "Evolutionary induction of sparse neural trees," *Evolutionary Computation*, 5(2): 213-236.