Proceedings of the 5th WSEAS Int. Conf. on SIMULATION, MODELING AND OPTIMIZATION, Corfu, Greece, August 17-19, 2005 (pp301-306)

Application of Cooperative Neural Network Ensemble for Developing Control System of Autonomous Robots

Md. Shahjahan, Y. Sakuragawa and K. Murase
Dept. of Human and Artificial Intelligence Systems
Faculty of Engineering, University of Fukui,
Bunkyo 3-9-1, Fukui shi 910-8507, Japan.
Email: jahan@synapse.his.fukui-u.ac.jp

Abstract: - A single neural network (SNN) is often used as a control system for an autonomous robot. We propose Cooperative neural network Ensemble to design the Control System (CECS) of an autonomous robot due to the inability of SNN on the way of capturing the real world environment. We describe simulations on populations of neural network (NN) ensembles in two different situations where single neural network (SNN) controller and simple neural network ensemble (SNNE) are not effective. Firstly, we report simulations using a set of SNN controller. Secondly, we report results that the proposed architecture produces functionally different groups of weights for different individual neural network (INN) in the controller by means of a correlation function. We minimize a correlation function during the course of evolution which produce functionally different INN and weight vector for each INN. The robot can clearly differentiate its movements, following its either left side or right side environment all the way around. We confirm the results by a temporal correlation map (TCM) of the outputs of multiple neural networks.

Key-Words: - Aritificial neural network, ensemble network, autonomous robot, evolution, learning

1 Introduction

There have been many attempts to evolve the neural controller of an autonomous robot [1, 2, 3]. These studies have been conducted with only evolution itself. In the last several years researchers have reported artificial evolution and learning techniques for studying the interaction between learning and evolution [4, 5, 6, 7]. It has been reported that learning boosts the evolutionary process. Genetic algorithms and evolutionary algorithms with fuzzy theory were also studied in optimization problems in robotics [8]. Unfortunately, none of the works on only the evolution and on the evolution with learning have addressed the use of neural network ensemble controller during the course of evolution.

The attempts so far to evolve the control systems of autonomous robots concern with many evolutionary approaches with a large variety of structure of neural networks. These include feedforward neural network, dynamic time recurrent networks, etc. However, most of them consist of single neural network. There are few limitations of single network: (i) they do not generalize well; (ii) they can not acquire much knowledge; (iii) even they acquire, the cross-talk might occur. The cross-talk is defined as indecision condition during the time of turning or classifying real world environment. (iv) weights are so distributive that one can not perceive the task learned by it. Although SNNE can partially solve those problems, still there is lack of interaction between INNs of entire ensemble. There is no benefit if each INN in the ensemble learns the same task. In addition, the problem of distributed weights remained unsolved. Therefore, we propose CECS architecture as controllers that solved the above problems significantly and effectively.

In the method, all individuals in a population are evolved according to fitness function and each individual has to learn a cooperation function between one generation to the other. The object of learning such correlation function is to decorrelate weights of control system as well as output of each component network in the control system. As a result, weights of one component network become functionally different from other component network and correlations among the involved networks become as negative as possible according to learned correlation. The correlations of outputs between two component network become also negative. As a result, one component network specilizes on the right movement and other on the left.

2 Evolution with Negative Correlation Learning

2.1 Evolutionary algorithm

Evolution and learning are two forms of adaptation that operate on different time scales [9]. Evolution is a process of selective reproduction and substitution based on the existence of a population of individuals displaying some variability. Learning, instead, is a set of modifications taking place within each single individual during its own life time [9]. A typical cycle of the evolution is named ***EvoA*** and has the following steps.

1. Create an initial population of N individuals where each of them represent a control system.
2. Decode each individual to a corresponding control system.
3. Allow a robot to perform evolutionary task for a life time.
4. Reproduce a number of children for each individual in the current generation based on fitness.
5. Apply genetic operators to the children generated above and obtain next generation.

2.2 Negative Correlation Learning

In order to implement the negative correlation learning at the individual level of entire population, we have to correlate the outputs of each network in the negative sense. The negative correlation learning in the ensemble network proposed by Liu & Yao (1999) [10] can not be directly applied to the robot controller, since we do not have teaching signal and we are not interested here to produce teaching signal due to huge computation. We minimize the correlation function only for each individual.

Start the first generation of the evolutionary process. Pick first chromosome (individual) to undergo correlation learning process. Allow it for several second to navigate in the environment. Record several sets of (10 set) the inputs and outputs. Update the weight of the corresponding chromosome according to recorded sets. Consider the Fig. 1. Let M and $F_i(n)$ is total number of networks and the output of the network i at the nth input sample, respectively. We define the average of the outputs as $F(n)$,

$$F(n) = \frac{1}{M}\sum_{i=1}^{M} F_i(n). \tag{1}$$

We minimize the same correlation function as the one proposed by Liu & Yao (1999) [10] and can be defined as $p_i(n)$ for the ith network at the nth input sample.

$$p_i(n) = (F_i(n) - F(n))\sum_{j\neq i}(F_j(n) - F(n)) \tag{2}$$

The partial derivative of $p_i(n)$ with respect to the output of network i on the nth input sample is

$$\frac{\partial p_i(n)}{\partial F_i(n)} = \sum_{j\neq i}(F_j(n) - F(n)). \tag{3}$$

$$= -2(F_i(n) - F(n)). \tag{4}$$

Thus in this learning the weight update rule become

$$\Delta w_i = -\lambda\frac{\partial p_i(n)}{\partial F_i(n)}.\frac{\partial F_i(n)}{\partial w_i} \tag{5}$$

$$= 2\lambda(F_i(n) - F(n))x_i \tag{6}$$

$$w_i = w_i + \Delta w_i \quad (7)$$

The parameter $\lambda > 0$ is the strength of learning. x_i is the sensory input to the network. Since sensory input x_i takes the values between 0-1023 and network output also more or less varies between 0-1023, we rescale the weight update from 0-15 in order to facilitate computation.

3 Experiments

In order to study the evolution with correlation learning, we used the method to develop a real autonomous mobile robot. In this section, we describe how it was implemented.

3.1 Mobile robot

A real mobile robot Khepera is used in this study. The structure and function of the robot has been well described elsewhere [11]. In short, the robot is equipped with eight infrared (IR) sensors (six in front and two in rare side) and two dc motors. These sensors work as proximity sensors for detecting object, by emitting infrared light and measuring its reflection. Several complete modules such as vision and gripper modules can be added to the basic structure.

3.2 Environment

The environment used to accomplish the evolutionary processes consisted of a square area with four obstacles as shown on the right of Fig. 1. The size of the area was approximately 60×60 cm. White non-glossy card-board was pasted on straight wooden bars. They were used to make the walls of the path and obstacles in the environment. The height of obstacles was made of short so that proximity sensors of the robot can detect them. The environment was illuminated from above by a 60-watt light bulb.

3.3 Control network and encoding

A simple two-layered feed-forward neural network was used as a component network of the entire ensemble. Two component networks were used as a control system to produce motor control signals as shown in the left of Fig. 1. Two control signals for motors were produced by summing the values from IR and/or vision sensors. That is, each output was generated by

$$S_m = S_b + G \sum w_i x_i \quad (8)$$

where, S_m, S_b, G, w_i, and x_i represent the output value to the motor, the base navigation speed of the motor, global gain, connection strength and input sensor signals value, respectively. The value of S_b and G were set to 5 and 1/1600, respectively. The global gain determines the sensitivity of to the modulation signal from sensors.

A direct coding scheme was used to encode an entire ensemble network. Each weight is encoded on a gene of five bits where the first bit determines the sign of the weight and the remaining four bits its strength. We use front six sensors. So each component network has a total of 12 weights. Thus for two component network a total of $12 \times 5 \times 2 = 120$ bits are necessary to represent a chromosome. The output signals to the motor are generated by averaging of the output of the component networks.

3.4 The task and fitness function

The simple task was given to the robot: navigate in the environment by avoiding obstacles. The evolution and correlation learning were carried out in the consecutive steps. That is, after each generation each individual was taken and used to navigate in the environment for a certain time. The process is repeated for all individuals and then evolutionary process enters into its 2nd generation.

We used the most widely used fitness function as defined below.

$$f = \sum_t V(t)(1 - \Delta v(t)) \left(1 - \sum_{i=1}^{n} s_i(t)\right). \quad (9)$$

here, V is the average rotation speed of two motors and is used to reward fast controllers. Δv is the absolute value of the algebraic difference between the signed speed values of the motors (one direction is positive and the other is negative) and is used to reward straight locomotion.

s_i is the proximity measure of the ith sensor and is used to punish the robot each time it sensed obstacles.

3.5 Experimental protocol

The evolutionary process should take place on the population level and learning should take place in individual level. Therefore, the method consists of two stage. In the first stage, evolution takes place on the entire population and in the 2nd stage correlation learning takes place in each individual. Since we do not have input-output data for correlation learning, we collected these data from the environment. In between two successive generations, one individual is taken into the environment and 5 second is allowed to travel into it. Within this 5 second, the robot recorded the 10 set of input and output data. After this, weight of the individual is updated using equation (6).

4 Results

This section reports the results and analysis of applying the ensemble architecture in the robot.

4.1 Evolutionary and learning process

4.1.1 Evolutionary process

Figure 2 (right) shows the evolutionary process of the proposed architecture. The average fitness of the population is improving in the course of evolution. Since we update the weight slightly after each generation, this does not affect the fitness function. The robot can easily navigate in the environment avoiding obastacles.

4.1.2 Temporal correlation map

Correlation learning process is shown in the left of Fig. 2. This is a temporal correlation map after each generation of all individual in a population. Cross-correlation was computed based on the output of the component networks. For each individual, we collected a series of 10 input-output data of two component networks and compute their correlation. Thus, at the end of each generation, there will be 10 correlation as plotted in the left figure as a function of generation. Therefore, we clearly observe that correlation becomes gradually negative at the latter generation due to the correlation term we used.

4.1.3 Functional weight

In order to see the influence of the correlation function, we record the weight distribution. As shown in Fig. 3 the weights of the two component networks are functionally different. In fact, they are negatively correlated. The movement of the robot is also distinguished by the two complementary functioning weights. Each component network is specialized on either its right side world or left side world. The robot can easily move with this two movements in the environment. The use of single neural network prohibits such kind of solution. In that case, two solution coexisting in a single network may mislead the right movement in the environment.

5 Discussion

In this paper, we integrate two techniques; evolution in the environment and correlation learning of individuals in a separate stage in between successive two generation. The novelty of the proposed method is to use the correlation function that plays a central role in separating the specialization of the different aspects of the real world on the control system of robot. The process in CECS is to put whole population into evolution and then into modification of individual weights slightly. This process repeats until a criteria is satisfied or a fixed number of generation is reached.

The CECS explores the specialization based learning through correlation learning during the course of evolution. This kind of specialization based learning is not possible for single two or three layer neural network. A single neural network reserves an average knowledge about the environment which may mislead on some slightly different sensory information. Unlike other attempts, CECS implements evolution with correlation learning which promotes the interaction of the networks and results spe-

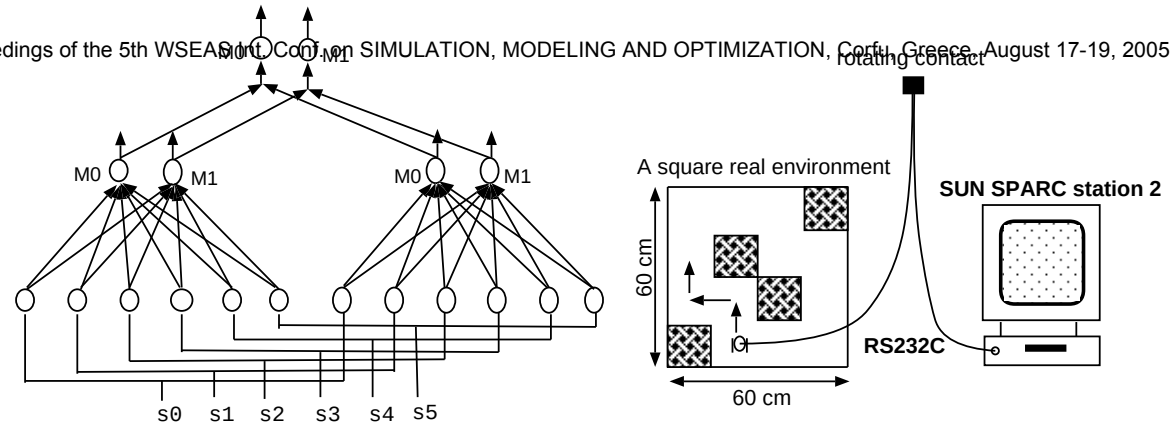

Figure 1: The proposed architecture and environment of the robot navigation

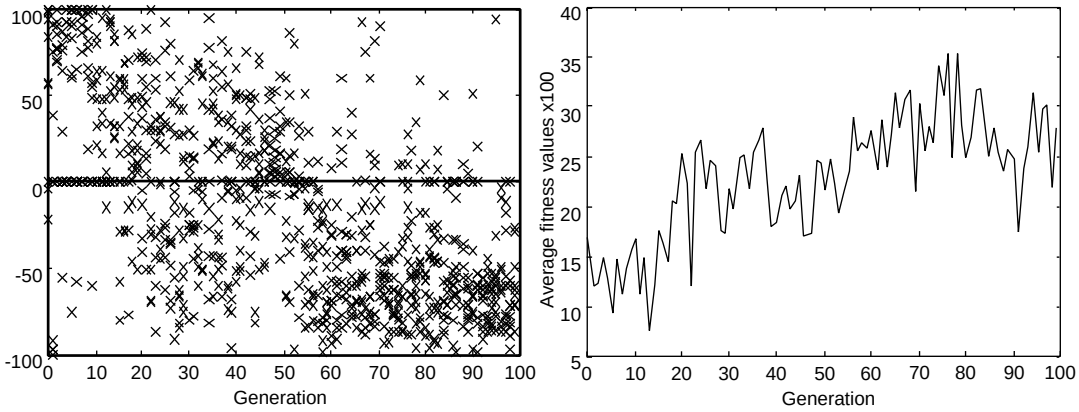

Figure 2: The cross-correlation of the network, computed only for right motor, as a function of generation. At each generation there are 10 cross-correlations for 10 individuals. The right figure plots fitness vs generation.

cialization based learning. The CECS is the first attempt in which we utilize evolution and correlation learning together, as of our knowledge. The parameter λ was set to 0.50.

Although we used a static environment, it is not strictly static, since the setup was in an open area where environmental condition may change. However, we need further experiments in order to deal with the external disturbance.

6 Conclusion

We describe simulations on a real mobile robot using ensemble neural network as a control architecture with correlation learning during the course of evolution. The use of negative correlation learning for the specialization based learning is well-established. The complementary representation of weights that exhibits approximately complementary functioning of movement and the negatively correlated component networks are appeared in the controller. The left and right outputs are approximately negatively correlated for almost all individuals. As a result, a component network of mobile robot controller experiences a complementary functioning with other one. That is one network become expert on its right side world and another one on its left side world.

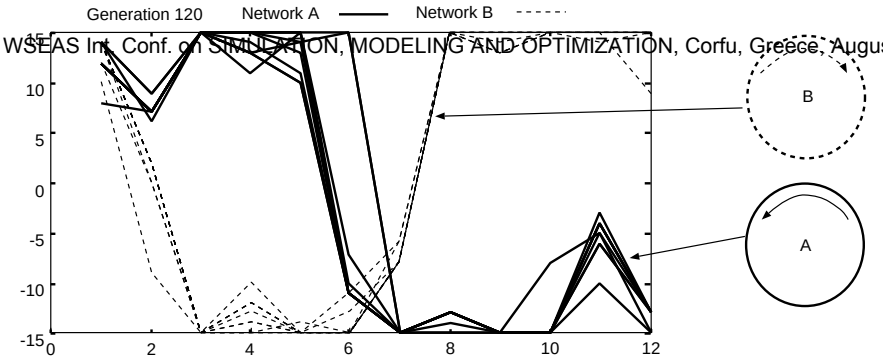

Figure 3: The converged weights and corresponding movement of the robot after 120 generation. The X-axes denotes the number of weights and Y-axes denotes the values of weights. The circles on the right side indicate the assumed movement of the robot.

References

[1] D. Floreano and F. Mondada, "Evolution of Homing Navigation in a Real Mobile Robot," *IEEE Transaction on Systems, Man, and Cybernetics* -Part B, Cybernetics 26(3), pp. 396-407, 1996.

[2] D. Floreano and F. Mondada, "Evolutionary Neurocontrollers for Autonomous mobile Robots," *Neural Networks,* 11, pp. 1461-1478, 1998.

[3] K. Sims, "Evolving 3D Morphology and Behavior by Competition," Artificial Life IV, Proceedings of the *Fourth International Workshop on the Synthesis and Simulation of Living Systems,* Ed. R. Maes, pp. 28-39, 1995.

[4] S. Nolfi and D. Floreano, "Evolutionary Robotics: Biology, Intelligence, and Technology of Self-Organizing Machines. *MIT Press, Cambridge, MA,* 2000.

[5] S. Nolfi, D. Floreano, "Learning to Adapt to Changing Environment in Evolving Neural Networks," *Adaptive Behavior*, 1, pp. 99-105

[6] B. Yamauchi and R. D Beer, "Sequential Behavior and Learning in Evolved Dynamical Neural Networks," *Adaptive behavior*, 2(3), pp. 219-246.

[7] R. D. Beer and J. C Gallagher, "Evolving Dynamic Neural Networks for Adaptive Behavior," *Adaptive Behavior*, vol. 1, pp. 91-122, 1992.

[8] T. Fukuda, N. Kubota, and T. Arakawa, "GA Algorithms in Intelligent Robots, in Fuzzy Evolutionary Computation, *Kluwer Academic Publishers*, pp. 81-105, 1997.

[9] S. Nolfi and D. Floreano, "Learning and Evolution," *Autonomous Robots*, 7(1), 1999.

[10] Y. Liu and X. Yao, "Ensemble Learning via Negative Correlation," *Neural Networks,* vol. 12, pp. 1399-1404, 1999.

[11] F. Mondada, E. Franzi, and P. Ienne, "Mobile Robot Miniaturization: A Tool for Investigating in Control Algorithm," Proc. of *Third International Conference on Experimental Robotics,* Kyoto, Japan, pp. 501-513, 1993.