An Artificial Neural Network for a Tank Targeting System

Hans W. Guesgen and **Xiao Dong Shi**
Computer Science Department, University of Auckland
Private Bag 92019, Auckland, New Zealand
hans@cs.auckland.ac.nz, eric0705@gmail.com

Abstract

In this paper, we apply artificial neural networks to control the targeting system of a robotic tank in a tank-combat computer game (RoboCode). We suggest an algorithm that not only trains the connection weights of the neural network, but simultaneously searches for an optimum network architecture. Our hybrid evolutionary algorithm (PSONet) uses modified particle swarm optimisation to train the connection weights and four architecture mutation operators to evolve the appropriate architecture of the network, together with a new fitness function to guide the evolution.

Introduction and Background

Artificial Neural Networks (ANNs) have been used in a variety of areas during the last thirty years (Meyer 1998; Russell & Norvig 2003; Scapura 1995), more recently in computer games to improve the quality of the artificial intelligence engine in these games (Schaeffer 2000). This paper discusses the application of ANNs to control the targeting system of a robotic tank in a tank-combat game, using the Robocode environment (Robocode 2005; Robowiki 2005) as a platform.

ANNs have the ability to learn over time and therefore to adapt to new situations and strategies. In general, the structure of an ANN determines its performance. Some traditional algorithms use a fixed structure and only train the weights of the connections to optimise the network. Others discover a relative optimum architecture first and then train the weights on this architecture (Koza & Rice 1991; Odri, Petrovacki, & Krstonosic 1993; Yao & Liu 1997). Since these algorithms are very prone to overfitting or convergence on local optima, we suggest to apply a hybridized and evolutionary algorithm (PSONet), which simultaneously finds the best structure for the ANN and optimal weights for its connections by using a new fitness function.

Methodology and Architecture

We restrict ourselves here to fully connected multilayer feedforward networks, i.e., neural networks in which information is passed from the input nodes through the hidden nodes to the output nodes. Theoretical results show that, given enough hidden nodes, such a network can approximate any reasonable function to any required degree of accuracy. This is usually achieved by training the network with an error backpropagation algorithm (Scapura 1995).

Since backpropagation just optimizes the weights of the connections on a predefined neural network architecture, it is difficult to avoid the underfitting or overfitting problem. An evolutionary approach (Salomon 1998), like particle swarm optimisation, can overcome these problems. Particle swarm optimisation (PSO) is a stochastic global optimization technique inspired by the social behavior of bird flocking (Kennedy & Eberhart 1995; Shen *et al.* 2004; Zhang, Shao, & Li 2000). The particles share information with each other, in particular information about the quality of the solutions they have found at specific points in the search space. The best solution discovered by a specific particle is referred to as the personal best solution. Particles move towards other personal best solutions with certain velocities in order to discover improved solutions.

We propose a modified particle swarm optimisation algorithm with an annealing factor combined with architecture mutation operators. The proposed approach optimises the connection weights and the architectures of the neural networks simultaneously and thereby avoids the problem of slow convergence speed and the tendency to overfitting. The particle swarm optimisation algorithm is used for training the weights of the neural networks, whereas the architecture mutation operators (hidden node deletion, connection deletion, connection addition, and hidden node addition) are applied to find the optimal network structure. The individual steps of our algorithm, called PSONet, are summarized in Figure 1.

The efficiency and quality of the algorithm depends significantly on the fitness function used to rank the neural networks. It is based on two factors: the prediction accuracy and the complexity of the network.

The accuracy of a neural network is defined by the root-mean-square error (RMSE):

$$RMSE = \frac{\sqrt{\sum_{i,j}(O_{ij} - T_{ij})^2}}{S \cdot N}$$

where O_{ij} and T_{ij} are the actual value and target value, respectively, for the jth output in the i training example. S is the size of training set and N the number of output nodes.

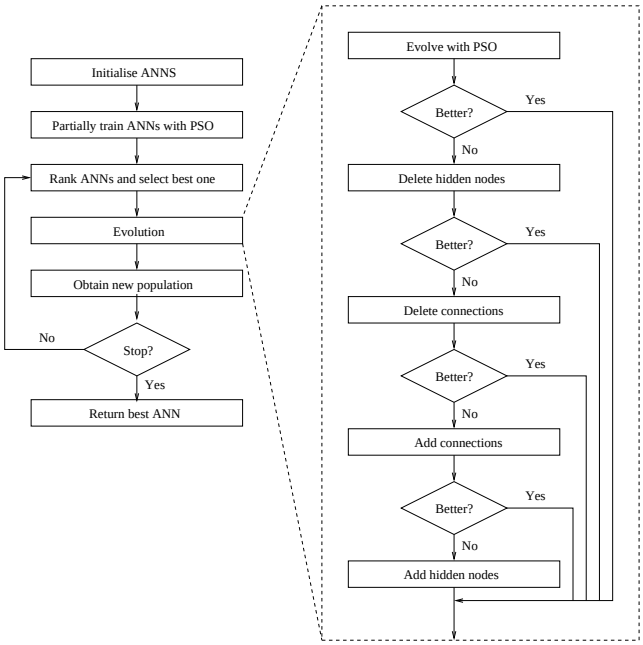

Figure 1: Scheme of the PSONet algorithm.

The complexity of a neural network can be expressed in terms of a penalty:

$$PENALTY = \zeta \cdot \frac{k}{K}$$

where k is either the number of connections with nonzero weights or the number of connections involved in evolution. K is the total number of connections of a neural network. ζ is a weighting factor that determines how much the complexity affects the performance of fitness function.

The overall fitness of a neural network is defined as:

$$FITNESS = RMSE \cdot (1 + PENALTY)$$

When the fitness of a neural network reaches a predefined threshold, the algorithm terminates (for the particular subpopulation). This is also the case if the number of generations has exceeded a maximum or if no improvement has been made after a certain time period.

Conclusion

The goal of the work described in this paper is to investigate the feasibility of using an evolutionary algorithm to evolve the targeting system (artificial neural networks) of a robot tank in a challenging and realistic battle environment (Robocode). Our approach uses architecture mutation operators to find the structure of the network and PSO to train the weights of its connections. Although PSO is a population-based optimisation algorithm, it does not employ any evolution operators (such as crossover or mutation). As a result, computation costs are lower and fewer parameters have to be adjusted. In addition, PSO converges quickly and avoids the overfitting problem to some extent.

	ENN/ SpinBot	ENN/ Iguana_2f45	ENN/ PheonixM
Total score	8208/934	7725/1313	5971/2372
Survival rate	100%	98%	90%

Table 1: Number of nodes and connections in the networks produced by from PSONet using the test robots for training.

The algorithm was tested in competitions with three different opponents, which we downloaded from the web: SpinBot, Iguana_2f45, and PheonixM. The three opponents that we chose for testing represent the most common strategies in Robocode. Ten runs were conducted for each moving strategy, resulting in different neural networks to control our tank. Table 1 shows the competition results between ENN and SpinBot, Iguana_2f45 and PheonixM, respectively. ENN is the robot that uses the evolved neural networks as its targeting system. Each competition includes 50 rounds.

References

Kennedy, J., and Eberhart, R. 1995. *The Particle Swarm: Social Adaptation in Information Processing Systems*. New York: McGraw-Hill.

Koza, J., and Rice, J. 1991. Genetic generation of both the weights and architecture for a neural network. In *Proc. IJCNN-91*, 397–404.

Meyer, J.-A. 1998. Evolutionary approaches to neural control in mobil robots. In *Proc. IEEE International Conference on Systems, Man and Cybernetics*, 35–40.

Odri, S.; Petrovacki, D.; and Krstonosic, G. 1993. Evolutional development of a multilevel neural network. *Neural Networks* 6:583–595.

Robocode. 2005. http://robocode.sourceforge.net/. Last accessed on 16 November 2005.

Robowiki. 2005. http://robowiki.net. Last accessed on 18 November 2005.

Russell, S., and Norvig, P. 2003. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, New Jersey: Prentice Hall, 2nd edition.

Salomon, R. 1998. Evolutionary algorithms and gradient search: Similarities and difference. *IEEE Transactions on Evolutionary Computation* 2:45–55.

Scapura, D. 1995. *Building Neural Networks*. Boston, Massachusetts: Addison-Wesley Professional.

Schaeffer, J. 2000. The game computers (and people) play. In Zelkowitz, M., ed., *Advances in Computers*, volume 50. San Diego, California: Academic Press. 189–266.

Shen, Q.; Jiang, J.; Jiao, C.; Lin, W.; Shen, G.; and Yu, R. 2004. Hybridized particle swarm algorithm for adaptive structure training of multilayer feed-forward neural network: Qsar studies of bioactivity of organic compounds. *Journal of Computational Chemistry* 25:1726–1735.

Yao, X., and Liu, Y. 1997. A new evolutionary system for evolving artificial neural networks. *IEEE Transactions on Neural Networks* 8:694–713.

Zhang, C.; Shao, H.; and Li, Y. 2000. Particle swarm optimization for evolving artificial neural network. In *Proc. IEEE International Conference on Systems, Man and Cybernetics*, 2487–2490.