The Effect of Noise on the Performance of Cultural Evolution in Multi-Agent Systems

Dara Curran
Dept. of Information Technology
National University of Ireland, Galway
Email: dara.curran@nuigalway.ie

Colm O'Riordan
Dept. of Information Technology
National University of Ireland, Galway
Email: colmor@geminga.it.nuigalway.ie

***Abstract*— This paper examines the effect of the addition noise to the cultural learning process of a population of agents. Experiments are undertaken using an artificial life simulator capable of simulating population learning (through genetic algorithms) and lifetime learning (through the use of neural networks). To simulate cultural learning, (the exchange of information through non–genetic means) a group of highly fit agents is selected at each generation to function as teachers which are assigned a number of pupils to instruct. Cultural exchanges occur through a hidden layer of an agent's neural network known as the verbal layer. Through the use of back–propagation, a pupil agent imitates the teacher's behaviour and overall population fitness is increased.**

We show that the addition of noise to cultural exchanges can improve on the performance of cultural learning.

I. Introduction

Culture is a means to exchange of information through non–genetic means. Examples of such exchanges are language, symbols and artifacts. Means of cultural exchanges between organisms may be considered a subset of lifetime learning, known as cultural learning. Generally, these exchanges allow more experienced organisms to impart some domain knowledge to less informed organisms.

Cultural exchanges in evolving populations can be simulated in computer systems using neural networks and genetic algorithms respectively. Genetic algorithms represent potential problem solutions as genetic codes which are then evaluated for fitness. Pairs of codes are selected in proportion to their fitness and are combined together to produce offspring. These offspring become part of the next generation and the process is repeated. Genetic algorithms have been shown to be useful in a vast variety of problem domains.

Neural networks are simplified mathematical models of nervous systems, inspired by the neurons and synapses of living creatures. Neural networks function by reading input patterns, feeding these pattern values through a succession of weighted synapses between neurons, and finally displaying output values at specified output neurons. By comparing a network's output pattern to the desired output pattern for a given input, a measure of error can be obtained. This error is then used to alter the weighting value of synapses that connect neurons in the network. One such algorithm for performing this weighting adjustment is known as error back propagation. Through a series of training iterations, the overall error of a network can be reduced,thereby improving its performance.

The combination of genetic algorithms and neural networks provides a framework for evolutionary experimentation in popular domains such as language evolution, neural network design optimization and games.

In previous work by Parisi *et al.* [1], it was suggested that the addition of noise to a teacher's verbal output could enhance a population's ability to retain culturally acquired information. This paper presents experiments that replicate these results and furthermore, attempt to ascertain the level of noise most suitable for the successful imparting of information.

The remainder of the paper is organised as follows: Section 2 gives some background information on related work. Section 3 presents the artificial life simulator employed in the experiments. Each of its components are discussed, genetic algorithm, neural network architecture and the encoding used to generate valid genetic codes from neural network structures. In Section 4, we detail the results and in Section 5, conclusions are presented and future directions are outlined.

II. Related Work

A number of learning models can be identified from observation in nature. These can roughly be classified into two distinct groups: population and life-time learning. In this paper we consider a subset of lifetime learning, cultural learning.

A. Population Learning

Population learning refers to the process whereby a population of organisms evolves, or learns, by genetic means through a Darwinian process of iterated selection and reproduction of fit individuals. In this model, the learning process is strictly confined to each organism's genetic material: the organism itself does not contribute to its survival through any learning or adaptation process.

B. Lifetime Learning

By contrast, there exist species in nature that are capable of learning, or adapting to environmental changes and novel situations at an individual level. Such learning, know as life-time learning, still employs population learning to a degree, but further enhances the population's fitness through its adaptability and resistance to change. Another phenomenon related to life-time learning, first reported by Baldwin [2], occurs when certain behaviour first evolved through life-time learning becomes imprinted onto an individual's genetic material through the evolutionary processes of crossover and mutation. This individual is born with an innate knowledge of such behaviour and, unlike the rest of the populations, does not require time to acquire it through life-time learning. As a result, the individual's fitness will generally be higher than that of the population and the genetic mutation should become more widespread as the individual is repeatedly selected for reproduction.

Research has shown that the addition of life-time learning to a population of agents is capable of achieving much higher levels of population fitness than population learning alone [3], [4]. Furthermore, population learning alone is not well suited to changing environments [5].

1) Cultural Learning: Culture can be succinctly described as a process of information transfer within a population that occurs without the use of genetic material. Culture can take many forms such as language, signals or artifactual materials. Such information exchange occurs during the lifetime of individuals in a population and can greatly enhance the behaviour of such species. Because these exchanges occur during an individual's lifetime, cultural learning can be considered a subset of lifetime learning.

An approach known as synthetic ethology [6], [7] argues that the study of language is too difficult to perform in real world situations and that more meaningful results could be produced by modelling organisms and their environment in an artificial manner. Artificial intelligence systems can create tightly controlled environments where the behaviour of artificial organisms can be readily observed and modified.

In particular, experiments conducted by Hutchins and Hazlehurst [8] simulate cultural evolution through the use of a hidden layer within an individual neural network in the population. This in effect, simulates the presence of a Language Acquisition Device (LAD), the physiological component of the brain necessary for language development, the existence of which was first proposed by Chomsky [9]. The hidden layer acts as a verbal input/output layer and performs the task of feature extraction used to distinguish different physical inputs. It is responsible for both the perception and production of signals for the agent.

A number of approaches were considered for the implementation of cultural learning including fixed lexicons [10], [11], indexed memory [12], cultural artifacts [13], [14] and signal–situation tables [6]. The approach chosen was the increasingly popular teacher/pupil scenario [15], [1], [11] where a number of highly fit agents are selected from the population to act as teachers for the next generation of agents, labelled pupils. Pupils learn from teachers by observing the teacher's verbal output and attempting to mimic it using their own verbal apparatus. As a result of these interactions, a lexicon of symbols evolves to describe situations within the population's environment.

III. Experiment setup

The experiments outlined in the next sections are designed to simulate an environment where agents may encounter food or poison bit patterns. Agents that correctly identify these patterns are awarded higher levels of fitness than agents who mistake food for poison.

For this set of experiments, food and poison bit patterns are 5-bit patterns representing the 5-bit parity problem, where food is assigned the value 1 and poison the value 0. The next section outlines the artificial life simulator that is employed in these experiments.

A. Simulator

The architecture of the simulator is based on a hierarchical model (figure 1). Data propagates from the simulator's interface down to the simulator's lowest level. The neural network and genetic algorithm layers generate results which are then fed back up to the simulator's highest level.

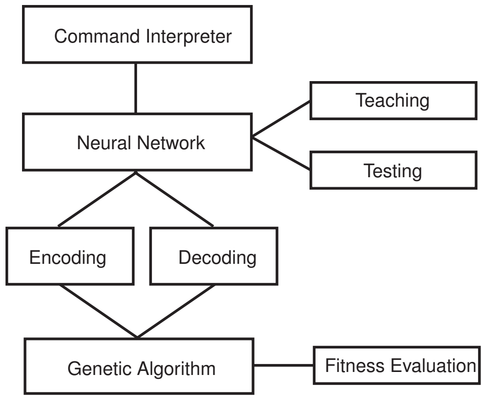

Fig. 1. *Simulator Architecture*

B. Command Interpreter Layer

The command interpreter is used to receive input from users in order to set all variables used by the simulator. The interpreter supports the use of scripts, allowing users to store parameter information for any number of experiments. A parsing algorithm is used to break down a user's commands and to pass appropriate instructions to the relevant components of the simulator.

C. Genetic Algorithm Layer

The genetic algorithm layers is responsible for the creation of successive generations of neural networks. The layer is responsible for the simulation of population learning within the simulator. The layer accepts data from the encoding layer as a genetic algorithm data structure. This structure is then manipulated to generate the next generation using the selection, crossover and mutation operators.

1) Selection: The selection process examines each individual in the population and must determine which individuals to place into the intermediate population. The simulator genetic algorithm layer uses the network's error and accuracy as a fitness measure to make this evaluation.

Once the fitness value is computed for each individual in the population, linear based fitness ranking is used to normalize fitness values evenly over the entire population producing fitness values in the range [0.0,0.5]. This is done to overcome the potential for stagnation across the population gene pool. Traditional proportional fitness assignment may produce a population with very similar fitness values, thereby deteriorating the selection process and possibly resulting in a loss of diversity stemming from a small number of individuals being allowed to reproduce many times. Ranking introduces a uniform scaling across the population's fitness so that fitness values are evenly spread and making the selection process more successful. In addition, fitness ranking provides a simple means of controlling selective pressure, the probability that the best individual is selected compared to the average probability of selection of all individuals [16].

The total number of individual networks in the population N_{ind} is sorted such that each individual occupies a ranking position pos where pos=1 represents the least fit individual and pos=N_{ind} the most fit individual. For selective pressure SP in [1.0,2.0], a network's fitness value $Fitness_{net}$is calculated as:

$$Fitness_{net} = 2 - SP + 2(SP - 1)(pos - 1)/(N_{ind} - 1)$$

Roulette wheel selection is then applied to the population to select individuals for the intermediate population. The number of times that an individual may be copied to the intermediate population is a parameter which can be set using the command interpreter.

2) Crossover: As a result of the chosen encoding scheme, crossover may not operate at the bit level as this could result in the generation of invalid gene codes. Therefore, the crossover points are restricted to specific intervals — only whole node or link values may be crossed over. Two–point crossover is employed in this implementation.

3) Mutation: The mutation operator introduces additional noise into the genetic algorithm process thereby allowing potentially useful and unexplored regions of problem space to be probed. The mutation operator usually functions by making alterations on the gene code itself, most typically by altering specific values randomly selected from the entire gene code. In this implementation, weight mutation is employed. The operator takes a weight value and modifies it according to a random percentage in the range [-200%,200%].

D. Encoding and Decoding Schemes

Before the encoding and decoding layers can perform their respective tasks, it is necessary to arrive at a suitable encoding scheme. Many schemes were considered in preparation of these experiments, prioritising flexibility, scalability, difficulty and efficiency. These included Connectionist Encoding[17], Node Based Encoding[18], Graph Based Encoding[19], Layer Based Encoding[20], Marker Based Encoding[21], Matrix Re–writing[22], [23], Cellular Encoding[24], Weight–based encoding[25], [26] and Architecture encoding[27].

The scheme chosen is based on Marker Based Encoding which allows any number of nodes and interconnecting links for each network giving a large number of possible neural network permutations.

	Start Marker	Node Label	Threshold	Link to Node	Link Weight	Link to Node	Link Weight	End Marker	
...	SM	5	0.8	4	0.83	3	-0.51	EM	...

Fig. 2. *Marker Based Encoding*

Marker based encoding represents neural network elements (nodes and links) in a binary string. Each element is separated by a marker to allow the decoding mechanism to distinguish between the different types of element and therefore deduce interconnections[22], [23].

In this implementation, a marker is given for every node in a network. Following the node marker, the node's details are stored in sequential order on the bit string. This includes the node's label and its threshold value. Immediately following the node's details, is another marker which indicates the start of one or more node–weight pairs. Each of these pairs indicates a back connection from the node to other nodes in the network along with the connection's weight value. Once the last connection has been encoded, the scheme places an end marker to indicate the end of the node's encoding (figure 2).

The scheme has several advantages over others:

- Nodes can be encoded in any particular order, as their role within the network is determined by their interconnecting links.
- The network structures may grow without restriction—any number of nodes can be encoded along with their interconnections.
- Links between nodes can cross layer boundaries. For instance, a node in the input layer may link directly to a node in the output layer, even if there are many layers between the two.
- The system encodes individual weighting values as real numbers, which eliminates the 'flattening' of the learned weighting values which can occur when real number values are forced into fixed bit–size number values.

E. Encoding and Decoding Layers

To perform genetic algorithm tasks, the neural network structures must be converted into gene codes on which the genetic algorithm will perform its operations. Conversely, once the genetic algorithm has performed its tasks, the genetic code structure must be converted back to a neural network architecture. The encoding and decoding layers must therefore accept both neural network data structures and genetic algorithm data structures to function correctly. The encoding and decoding layers follow the scheme outlined in Section 3.4.

F. Neural Network Layer

The neural network layer is responsible for all functions carried out by the neural networks in the simulator's population. The neural network layer accepts a population of neural network data structures and performs a number of functions.

G. Simulating Cultural Evolution

In order to perform experiments related to cultural evolution, it was necessary to adapt the existing simulator architecture developed by Curran and O'Riordan [3] to allow agents to communicate with one another. This was implemented using an extended version of the approach adopted by Hutchins and Hazlehurst. The last hidden layer of each agent's neural network functions as a verbal input/output layer (figure 3).

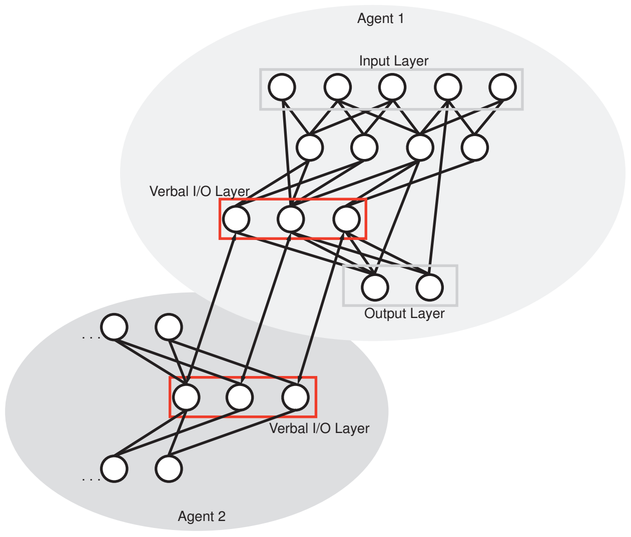

Fig. 3. *Agent Communication Architecture*

At end of each generation, a percentage of the population's fittest networks are selected and are allowed to become teachers for the next generation. The teaching process takes place as follows: a teacher is stochastically assigned n pupils from the population where $n = \frac{N_{pop}}{N_{teachers}}$, where N_{pop} is the population size and $N_{teachers}$ is the number of teachers. Each pupil follows the teacher in its environment and observes the teacher's verbal output as it encounters what it believes to be food or poison bit patterns. The pupil then attempts to emulate its teacher's verbal output using back-propagation. Once the teaching process has been completed, the teacher networks die and new teachers are selected from the new generation.

Unlike previous implementations, the number of verbal input/output nodes is not fixed and is allowed to evolve with the population, making the system more adaptable to potential changes in environment. In addition, this method does not make any assumptions as to the number of verbal nodes (and thus the complexity of the emerging lexicon) that is required to effectively communicate.

IV. Experiments

The purpose of this set of experiments is to identify the effect of noise on the performance of cultural learning. Noise is added to a teacher's output by a random value in the range [-0.5,0.5]. The probability of a cultural transmission being distored by noise is varied from 0.0 to 1.0. Other parameters for the cultural learning setting are chosen as follows: the teacher ratio, that is the percentage of the population that is chosen as teachers, is set at 0.1. The number of teaching cycles, the exposure each pupil has to the teachings of a particular teacher is set at 5 cycles. These parameters where chosen following a series of preliminary experiments. The crossover and mutation rates were set at 0.6 and 0.02 respectively. These values were determined empirically to give the best results. The results presented below are averaged from twenty experiment runs over 250 generations.

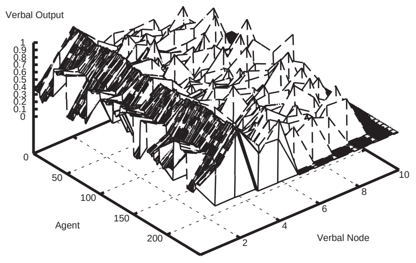

Fig. 4. *Verbal Output at Generation 1*

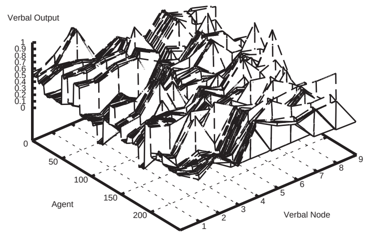

Fig. 5. *Verbal Output at Generation 125*

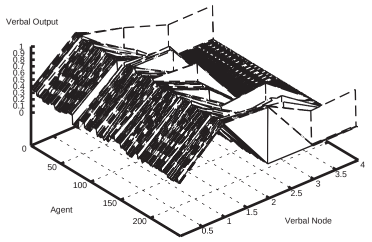

Fig. 6. *Verbal Output at Generation 250*

Figures 4 to 6 show the progression of the population's lexicon during one of the experiment runs for one of the 5–bit parity states. The figures show that a relatively random collection of verbal output evolves over time to an accepted lexical standard. By generation 250, most of the agents are using the same verbal outputs to describe their environment.

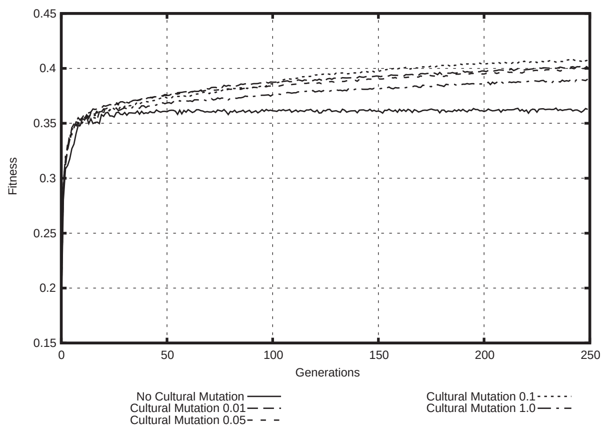

Fig. 7. *Cultural Mutation*

The results shown in figure 7 suggest that when cultural mutation is set at *any* of the values attempted, the population fitness improves. Even very high mutation (i.e. where all cultural exchanges are distorted by a value in the range [-0.5,0.5]) produces better results than no mutation at all. However, smaller amounts of cultural mutation produce higher levels of fitness. For this problem domain, it is clear from the results that the optimum level of cultural mutation is approximately 0.1.

Evidently, the diversity provided by the distortion of teacher output seems to give rise to better learning on the part of the pupils. We speculate that since the teacher's output is never perfect, and is never perfectly perceived by its pupil, it cannot be wholly relied upon. The addition of noise may help the pupil determine its own response to stimulus, rather than blindly copying the teacher's. Thus, the addition of noise adds an important element to the cultural learning process.

V. Conclusion

These experiments have shown that the addition of noise to cultural learning exchanges within a population of agents can greatly improve their overall fitness for this particular problem set. While it would be tempting to generalise on the effect of cultural mutation as a whole, it is clear that more analysis is required. We have expanded on previous work by showing that the optimum noise probability value can be ascertained for a given problem domain. Further work will focus on the co–evolution of the cultural mutation parameter to provide added plasticity in changing environments.

References

[1] D. Denaro and D. Parisi. Cultural evolution in a population of neural networks. In *M.Marinaro and R.Tagliaferri (eds), Neural Nets Wirn-96.New York: Springer*, pages 100–111, 1996.

[2] J.M. Baldwin. A new factor in evolution. In *American Naturalist 30*, pages 441–451, 1896.

[3] D. Curran and C. O'Riordan. On the design of an artificial life simulator. In V. Palade, R. J. Howlett, and L. C. Jain, editors, *Proceedings of the Seventh International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES 2003)*, University of Oxford, United Kingdom, 2003.

[4] D. Curran and C. O'Riordan. Artificial life simulation using marker based encoding. In *Proceedings of the 2003 International Conference on Artificial Intelligence (IC-AI 2003)*, volume II, pages 665–668, Las Vegas, Nevada, USA, 2003.

[5] D. Curran and C. O'Riordan. Lifetime learning in multi-agent systems: Robustness in changing environments. In *Proceedings of the 14th Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2003)*, pages 46–50, Dublin, Ireland, 2003.

[6] B. MacLennan and G. Burghardt. Synthetic ethology and the evolution of cooperative communication. In *Adaptive Behavior 2(2)*, pages 161–188, 1993.

[7] L. Steels. The synthetic modeling of language origins. In *Evolution of Communication*, pages 1–34, 1997.

[8] E. Hutchins and B. Hazlehurst. How to invent a lexicon: The development of shared symbols in interaction. In N. Gilbert and R. Conte, editors, *Artificial Societies: The Computer Simulation of Social Life*, pages 157–189. UCL Press: London, 1995.

[9] N. Chomsky. On the nature of language. In *Origins and evolution of language and speech*, pages 46–57. Annals of the New York Academy of Science, New York. Vol 280, 1976.

[10] H. Yanco and L. Stein. An adaptive communication protocol for cooperating mobile robots, 1993.

[11] C. Angelo and D. Parisi. The emergence of a language in an evolving population of neural networks. *Technical Report NSAL–96004, National Research Council, Rome*, 1996.

[12] L. Spector and S. Luke. Culture enhances the evolvability of cognition. In *Cognitive Science (CogSci) 1996 Conference Proceedings*, 1996.

[13] E. Hutchins and B. Hazlehurst. Learning in the cultural process. In *Artificial Life II, ed. C. Langton et al.* MIT Press, 1991.

[14] A. Cangelosi. Evolution of communication using combination of grounded symbols in populations of neural networks. In *Proceedings of IJCNN99 International Joint Conference on Neural Networks (vol. 6)*, pages 4365–4368, Washington, DC, 1999. IEEE Press.

[15] A. Billard and G. Hayes. Learning to communicate through imitation in autonomous robots. In *7th International Conference on Artificial Neural Networks*, pages 763–738, 1997.

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

[17] R. K. Belew, J. McInerney, and N. N. Schraudolph. Evolving networks: Using the genetic algorithm with connectionist learning. In Christopher G. Langton, Charles Taylor, J. Doyne Farmer, and Steen Rasmussen, editors, *Artificial Life II*, pages 511–547. Addison-Wesley, Redwood City, CA, 1992.

[18] D. W. White. *GANNet: A genetic algorithm for searching topology and weight spaces in neural network design*. PhD thesis, University of Maryland College Park, 1994.

[19] J. C. F. Pujol and R. Poli. Efficient evolution of asymmetric recurrent neural networks using a two-dimensional representation. In *Proceedings of the First European Workshop on Genetic Programming (EUROGP),*, pages 130–141, 1998.

[20] M. Mandischer. Representation and evolution of neural networks. In R. F. Albrecht, C. R. Reeves, and N. C. Steele, editors, *Artificial Neural Nets and Genetic Algorithms Proceedings of the International Conference at Innsbruck, Austria*, pages 643–649. Springer, Wien and New York, 1993.

[21] D. Moriarty and R. Miikkulainen. Discovering complex othello strategies through evolutionary neural networks. *Connection Science*, 7(3–4):195–209, 1995.

[22] H. Kitano. Designing neural networks using genetic algorithm with graph generation system. In *Complex Systems, 4, 461-476*, 1990.

[23] G. F. Miller, P. M. Todd, and S. U. Hedge. Designing neural networks using genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms and Their Applications*, pages 379–384, 1989.

[24] F. Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Centre d'etude nucleaire de Grenoble, Ecole Normale Superieure de Lyon, France, 1994.

[25] R. S. Sutton. Two problems with backpropagation and other steepest-descent learning procedures for networks. In *Proc. of 8th Annual Conf. of the Cognitive Science Society*, pages 823–831, 1986.

[26] J. F. Kolen and J. B. Pollack. Back propagation is sensitive to initial conditions. In Richard P. Lippmann, John E. Moody, and David S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 3, pages 860–867. Morgan Kaufmann Publishers, Inc., 1991.

[27] J. R. Koza and J. P. Rice. Genetic generation of both the weights and architecture for a neural network. In *International Joint Conference on Neural Networks, IJCNN-91*, volume II, pages 397–404, Washington State Convention and Trade Center, Seattle, WA, USA, 8-12 1991. IEEE Computer Society Press.