Proceedings of the International Symposium on Evolving Intelligent Systems, Plamen Angelov, Dimitar Filev, and Nikola Kasabov (Eds.), at the AISB 2010 convention, 29 March – 1 April 2010, De Montfort University, Leicester, UK

Application of ANN-GA Hybrid to run a conveyor control system

Paul Morley*
Department of Design & Innovation
The Open University
Milton Keynes, MK7 6AA, England
pm4958@tutor.open.ac.uk

Jeff Johnson
Department of Design & Innovation
The Open University
Milton Keynes, MK7 6AA, England

ABSTRACT
This paper describes the use of a genetic algorithm to find the link weights of an Artificial Neural Network used to control an industrial conveyor system, and further proposes a control strategy to allow continual adaptation of the system to changing circumstances thus maintaining the optimization of the system. Results indicate feasibility of this approach, and that performance of the ANN can be as good as conventional systematic controls

Key words
artificial neural networks, evolutionary computation, machine learning, genetic algorithm, self-adapting controls

1. INTRODUCTION
Artificial neural networks (ANNs), and genetic algorithms (GAs), are both well known and widely used in artificial intelligence fields, including their use in controls applications.

ANNs are useful for pattern recognition, and generalizing patterns in complex multi-dimensional space. GAs are useful for searching widely in large solution spaces.

The problem with ANNs, is finding the weights of the links. Various methods have been tried, the most widely known being back-propagation, which is essentially an error minimization (hill climbing) algorithm. However, there is a key limitation of this approach: it requires a set of training data which is already classified (in order to present an input pattern with a required output).
In an industrial context, conventional control systems are often set-up once achieving a reasonable level of performance (not necessarily optimal), and then left forever. However circumstances change, and in the context described here the mix of work coming through the system will drift causing a gradual fall-off in performance. Therefore it was considered that an learning system which could constantly adapt to the circumstances would be beneficial.

This paper explores the use of a hybrid system for controlling an industrial conveyor system. A single ANN is used as a black-box decision maker about the next operation to be executed, and a GA is used to find the link weights of a population of ANNs.

2. NEURAL NETWORKS
Neural Networks were first developed by McCulloch & Pitts, inspired by biological neural systems. The idea that of computing using a large number of relatively simple units working in parallel as opposed to the conventional paradigm of sequential computing is intuitively attractive.

There is also proof that in theory at least, a 2 layer network is able to implement any measurable function to an arbitrary degree of accuracy. This important result was first established by Kolmogorov (the Kolmogorov existence theorem is actually not very useful in practice, because of the unknown nature of the functions used by each unit) and then in a manner more directly relevant to ANN development by Hornik et al. [1]. Although these results show that a 2 layer (i.e. 1 hidden layer) network could in theory implement any function, in practice it may be the case that a more efficient implementation could be found by using more layers.

There are many forms of ANNs. In this experiment the a fairly simple feed-forward network was used with 3 processing layers. This network comprises layers of individual units fully interconnected to the next layer, each unit implementing a non-linear sigmoid function of a weighted sum of all the inputs

Traditionally the weights associated with the links between each neuron are found by a back-propagation algorithm, which is essentially an error minimization (hill climbing) algorithm. Whilst this is undoubtedly effective, it assumes that there is a Training Set of pre-classified data samples which can be used to train the network (set the weights) and then a separate testing set. This is known as Supervised learning. [2]

3. GENETIC ALGORITHMS

GAs provide a powerful way of exploring a complex solution space. Essentially a GA depends on being able to describe a system by a sequence of symbols - by analogy: a chromosome. Different system's chromosomes can be split and combined to create a new generation. Some form of fitness function is then used to select the 'best' individuals systems and these go forward to create the next generation and so on. A random operation is also usually introduced analogous to genetic mutation.

In essence, Genetic Algorithms tend to include the following features;

- a population of multiple instances of trial solutions to the problem in hand
- some way of determining the relative fitness or effectiveness of each solution
- some operator(s) which produce new solution instances based on the more effective previous instances.

There are lots of variations to this approach; the operators used can vary - for example, mutation & recombination are two common operators inspired by biological genetics, but are not the only ones possible. Equally, there are many different approaches for managing the population - the strongest instances in one generation may be retained in the next, or the entire population may be replaced with new instances - to name just two possibilities.

A GA will not guarantee to find the optimum solution, or even any solution. However, they tend to be effective in homing in on some effective solution, and are especially useful in extremely large search spaces. They are often called evolutionary algorithms; and although the analogy with biological evolution is strong, the actual mechanisms of biological evolution are far more subtle than these approximations.

4. HYBRID METHODS

Genetic Algorithms have been applied to both the problems of finding link weights of an ANN, and finding the an effective structure of a network with researchers reporting schemes which alternatively evolve only the structure of the network, leaving the problem of finding the weights to a deterministic approach - such as classical back-propagation: a gradient descent optimization algorithm, or keeping the structure fixed and evolving the weights. [3] provides a general review of this field.

For example, [4] describes an algorithm used to evolve both the weights and the structure of ANNs. See [5] for an earlier example which also used evolutionary programming to evolve both the weights and structure. This latter refers specifically to a system where the selection process acted on only the output of the algorithm and not on the ideas underlying the output. That this is effective and efficient is contrary to the view of [6]. [7] and [8] give further examples of the use of an GA with selection based only on overall performance.

5. INDUSTRIAL CONTEXT

The system was applied to control a conveyor system in the context of an industrial laundry to a set of dryers. The general layout is shown in figure 1.

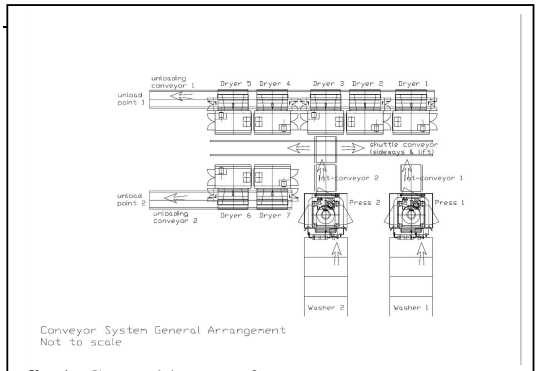

fig 1: General layout of conveyor system

System Description

This is a commercial laundry installation processing up to 6000kg of hospital linen per hour. The system was commissioned in November 2005, in a private facility in Ireland.

Batches of work move through the system, from the washers, to the water-extraction presses, one-batch storage conveyor, and into one of the dryers. The performance is affected by the shuttle conveyor which takes batches of work from one of the wash-lines, and takes them to one of the dryers. Performance is affected due to the transit time of the shuttle - for example:

- if the shuttle is in position for dryers 5 & 6, and a load of linen becomes available from washer 1, then washer 1 cannot unload until the shuttle has moved to that position thus losing that time for the washer.
- some types of work take longer in the dryers than others. If a shuttle puts long-drying work into a dryer close to the washers, then it ties up that dryer for a long time, hence dryers further away have to be used leading to longer transit times.

Generally best performance is achieved by putting longer dry-time loads in dryers further from the washers, and then parking the shuttle in front of the next washer to unload ready for it. This is the basis of conventional control algorithm. More subtle effects still cause sub-optimal performance, such as the tie-up

of the dryer unload conveyors caused by loads from - for example - dryers 1, 2 & 3 holding up the unloading of dryers 4 & 5.

Optimal utilization would achieve 30 batches from each washer per hour. Conventional control typically achieves 85-90% of this. The mix of work (with different proportions of batches with different dry times) varies over time and this causes a variation of performance.

6. HYBRID CONTROL ARCHITECTURE

A new control paradigm is proposed which has two phases:

- initial set-up
  A population of ANNs is generated and a GA is used to evolve to the point where the most successful ANN can be used to run the system. This is analogous to initial commissioning of the system and can be done quite quickly simulating the system on a computer.

  This paper describes work on this phase only.

- adaptive running
  The controlling ANN runs the system but a population of alternative ANNs is maintained evolving with live data. When an ANN from this population consistently achieves better results than the current controlling ANN, it supplants it. In this way the control will constantly adapt to changing circumstances

Some advantages of this architecture over conventional controls are;

- conventional controls require a full analysis of the operation of the controlled system. In this case this is possible but may not always be so
- conventional controls do not adapt to changing circumstances (the initial analysis may not hold for changed circumstances)
- the method is relatively easy to implement, and also takes very little adaptation to apply to control systems in other contexts.

However the disadvantages include;

- optimal control may not be attained, and without the system analysis, the optimal performance may not be known
- the operations of the system cannot be easily explained, could not be guaranteed so would not be suitable for safety critical systems

7. APPLICATION

A population of 50 ANNs was generated, each with 2 hidden layers of 28 & 13 units. The inputs to the ANN were parameters representing the state of the system - for example - a number representing the current load in the press, on the conveyor etc.

Each ANN was represented as a sequence of its link weights, and the GA was applied to this sequence on the following algorithm;

- run each ANN in turn for a simulated 2 hours operation and measure performance (loads produced)
- always keep best performing ANN
- all ANNs with less than average performance, eliminate from the population
- replace eliminated ANNs with new one generated by single point crossover from two parent ANNs each with better than average performance. Also apply a small mutation factor.

Fitness Function

To determine the performance of the ANN, a simple measure of how many batches the system had produced in the timescale was used. This method of measuring is remote from the individual decision outputs from the ANN. The key point is that the optimum output of the ANN for each individual decision made is not known without encountering the disadvantages discussed earlier. However the overall production of the system is known and furthermore is the parameter the operator of the system really needs to maximize.

Note that if the optimum output of the ANN was known for each decision, then simple gradient based learning would be appropriate. However, the optimum output could only generally be obtained following a full analysis of the system and this carries the disadvantages mentioned earlier.

8. ISLANDS

It was decided to implement an island strategy for the GA. This follows the parallel population strategy of [10]. This was carried over several phases;

phase 1 5 populations of 50 ANNs were evolved in isolation for 100 generations, with differing mutation rates in each population

phase 2 2 populations of 50 ANNs were evolved in isolation for 100 generations. Each population was formed from individuals mixed from the phase 1 populations

phase 3 1 population of 50 ANNs taken from a mix from the phase 2 populations. Evolution was run for 400 generations.

phase 4 the phase 3 generation was run for a further 500 generations, with a mutation rate steadily declining from 0.5% down to 0.1%

9. RESULTS & DISCUSSION

Figure 2 shows the increase in performance over the series, with a measure taken at the end of each set of 100 generations. The two curves show the performance of the best individual network in the final generation of the run, and the average of all networks over the final generation.

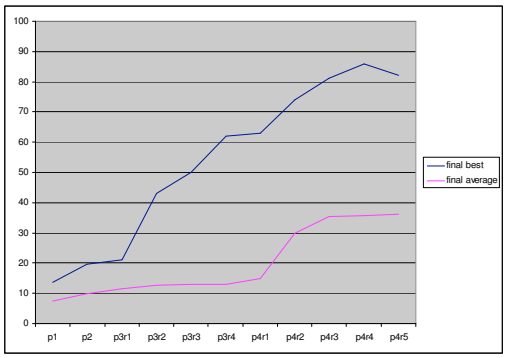

fig 2: performance (number of batches produced in the test time) of final best and final average ANNs for each evolutionary phase

The vertical axis shows the number of batches produced, over the trial period which was 5400 seconds (1.5 hours). The theoretical maximum production in this time is 90 batches It can be seen that after the various phases of evolution, the best performing networks are able to reach performances of well over 80, which is a very respectable system utilization. (Conventional control typically achieves 85-90% of this, or 76-81 batches).

Care must be taken not to overstate these results - although the evolved controls achieve a best here of 85 batches, on occasions (due to the mix of work coming through the system) the conventional control can achieve the maximum 90 batches. This is a stochastic system. However this does indicate the credibility of evolution for generating ANN system controls, which can be comparable to conventional methods.

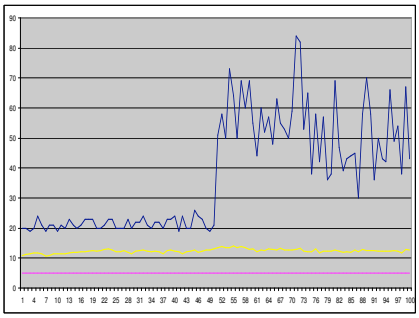

fig 3: performance (number of batches produced in the test time) of best, average, and worst ANNs.

Figure 3 shows the performance of the system during the 100 generations of phase 3 run 2. The top curve shows the best performing network in each generation. The lower curves show the average performance and the worst performance respectively. There is a step change from generation 50, (best performing network produced 21 batches), and generation 51, (best performing network produced 51 batches). Prior to this the performance had been relatively static. After this point, the best-performing network performance was relatively stable while the average performance climbed - as the successful elements of the best performing network propagated through the population. It is clear that the GA had jumped to a new and better part of the search space.

This particular result is a striking illustration of the random nature of evolutionary strategies and their ability to jump into a new area of the search space avoiding local minima.

10. REFERENCES

[1] Hornik K, Stinchcombe M, White H, 'Multilayer Feedforward Networks are Universal Approximators', Neural Networks 2 pp359-366, 1989

[2] Picton P, Introduction to Neural Networks, Macmillan, 1994.

[3] Yao X, 'Evolving Artificial Neural Networks', Proc. IEEE. Vol. 87 no. 9. 1999

[4] Curran D, O'Riordan C, 'Evolving Connect-Four Playing Neural Networks Using Cultural learning'. 2004. http://ww2.it.nuigalway.ie/cirg/publications.html

[5] Fogel D, 'Using Evolutionary Programming to Create Neural Networks that are Capable of Playing Tic-Tac-Toe'. Proc. Amer. Power Conf. 1993 pp875-879

[6] Penrose R, 'The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics'. Penguin Books. 1989

[7] Morley P, ' training a genetic algorithm and neural network hybrid pattern classifier by population statistics', AISB conf. 2005

[8] Morley P, 'evolving neural networks to play noughts & crosses', ISKE conf. 2009.

[9] Wolpert D H and Macready W G, "No free lunch theorems for optimization," IEEE Trans. Evolutionary Computation, vol. 1, pp. 67–82, Apr. 1997

[10] Whitley D, Rana S, and Heckendorn R. 'Island Model Genetic Algorithms & Linearly Separable Problems'. Journal of Computing and Information Technology. 1998.