

# Statistical Factors in Behaviour Learning

*Chris Thornton*

Cognitive and Computing Sciences  
University of Sussex  
Brighton  
BN1 9QH  
UK

Email: Christopher.Thornton@firenet.uk.com

Tel: (44)1273 678856

May 21, 2003

## Abstract

Various researchers have looked at ways of using supervised learning (i.e., training) to obtain adaptive robotic behaviours, e.g., [1,2]. However, the limitations of this approach are still unclear. This paper presents the results of an empirical study involving three behaviours and three, well-known learning algorithms. The results of the study suggest that ordinary supervised learning algorithms (such as ID3 [3] and Backpropagation [4]) only perform well when applied to a certain class of behaviours. A statistical analysis is presented which clarifies the characteristics of this class.<sup>1</sup>

## 1 Introduction

The paper presents an empirical study which investigated three behaviours:

- *Obstacle-avoidance*: the ability to move around an environment avoiding any obstacles.
- *Pursuit*: the ability to follow a moving object around an environment.
- *Foraging*: the ability to move towards relatively small objects and away from relatively large ones.

These behaviours were investigated using computer simulations.<sup>2</sup> The simulations involved a 2-dimensional world and very simple, simulated robots (or ‘animats’). The assumption was that the simulated robots had two free-wheeling

---

<sup>1</sup>A video of the simulations performed in the study is available from the author.

<sup>2</sup>The simulations were run under the Poplog environment running on a Sun SPARCstation 1+.

castors situated fore and aft and two drive wheels situated along the central, latitudinal axis. The aim was to get the algorithms to learn to control the speed of the two drive wheels so as to give the robot the desired behaviour (i.e. to move forwards or turn to right or left).

In all the simulations performed the simulated robots were able to probe their environments using an active ranging system. In each time step, the simulated robot's ranging system delivered a set of inputs (stimuli) showing the range to the nearest obstacle along a fixed set of 'rays'. The plan view shown in Figure 1 depicts a single simulated robot situated roughly in the centre of the space. The robot is represented as a small box with an arrow pointing in its direction of motion. The four dashed lines are the probe rays. The small integers show the points at which the probe rays have intersected with an obstacle.<sup>3</sup>

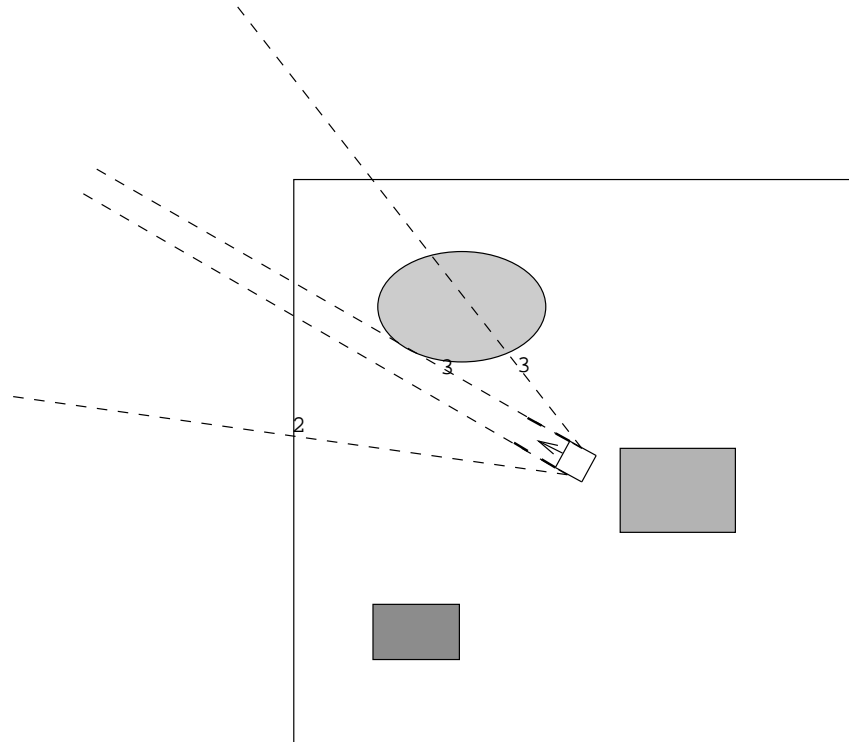


Figure 1:

---

<sup>3</sup>The actual number shown is the internal code used to represent the obstacle.

## 2 Training algorithms

The study covered the following supervised learning algorithms: ID3 [3], Back-propagation [5], Nearest-neighbours [6], Conjugate-gradient descent [7], Quick-prop [8] and Cascade-correlation [8]. Extensive trials showed that on the tasks considered the various algorithms produced comparable levels of generalization performance. Due to this lack of variation only a subset of the results are actually shown below. These give figures for a typical ‘connectionist’ algorithm (‘conjgrad’ or Conjugate-gradient descent) a typical ‘classical’ algorithm (ID3) and a typical ‘statistical’ algorithm (Nearest-neighbours).

## 3 Methodology

For each possible combination of learning algorithm and behaviour the following steps were carried out.

- (1) A ‘hand-simulation’ of the relevant behaviour (based on a hand-coded control procedure) was implemented.
- (2) In each step of the simulation, the robot’s stimulus-response profile was recorded.
- (3) The recorded stimulus-response pairs were presented as a training set to the learning algorithm.
- (4) A new simulation was run with the simulated robot being driven by the learned representation (i.e., the output from the learning algorithm).
- (5) The produced behaviour was evaluated as a reproduction of the original behaviour.

The first behaviour examined was ‘obstacle avoidance’ (just called ‘avoidance’ below). This behaviour involves moving around an environment avoiding any obstacles. In Figure 2 we see a short trace of a simulated robot producing avoidance behaviour. The robot’s position in each simulation step is shown as a small, arrow-topped box as before. Thus the sequence of boxes shows the robot’s trajectory around the environment. Note how the trajectory steers clear of all the obstacles. The simulated robot in this task received inputs from four range detectors arranged in a 45 degree arc.<sup>4</sup> The second behaviour examined was ‘pursuit’. In this behaviour the simulated robot must track a moving object around the environment. (The environment contained no obstacles in this case.) In Figure 3 we see a trace of a simulated robot producing the relevant behaviour. The simulated robot is shown here using dashed lines. The target is shown using unbroken lines. The simulated robot in this task received inputs from seven range detectors arranged in a 70 degree arc. The final behaviour

---

<sup>4</sup>In fact, in this simulation the robot used two pairs of ‘wing-mounted’ range detectors.

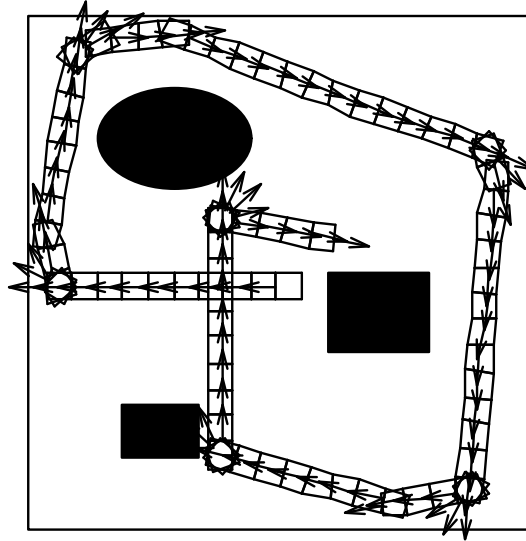


Figure 2:

examined was ‘foraging’. In this behaviour the simulated robot must move towards any small obstacles in the environment but *away* from any large ones. A typical scenario for the foraging behaviour is shown in Figure 4. There are two objects in the environment: one small and one large. The simulated robot in this task received inputs from seven range detectors arranged in a 100 degree arc.

## 4 Derived training pairs

As mentioned above, training sets for the learning algorithms were derived simply by writing down the robot’s stimulus-response profile at each step of the simulation. The inputs from the range detectors were presented to the robot as real numbers in the range 0-1. The inputs varied linearly and inversely with the measured distance. The amount of drive applied to the two wheels in each simulation step was stored in the form of two numbers in the range 0-1. Thus, a right turn with no forwards motion would be represented by the pair  $\langle 1 \ 0 \rangle$ . A sample of training pairs derived for the avoidance task is shown in Table 1. Note that the first four numbers in each row (training pair) are the range measures along the four rays. The final two numbers in each row specify the required amount of drive to be applied to the two drive wheels.

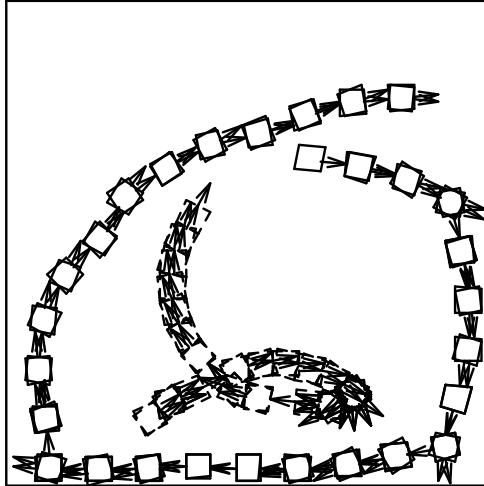


Figure 3:

0.95	0.94	0.79	0.79	1	0
0.91	0.29	0.29	0.22	1	1
0.92	0.93	0.91	0.88	1	0
0.65	0.6	0.59	0.19	1	1
0.94	0.93	0.9	0.84	1	0
0.7	0.55	0.54	0.13	1	1

Table 1:

## 5 Results

The results of the study are summarized in Table 2. The figures in the row labelled ‘hand-sim’ show the performance of the simulated robot during hand-simulation. (These are ‘optimal’ performance figures.) The figures in the row labelled ‘random’ show the performance obtained using a random move generator. The row labelled ‘conjgrad’ shows the performance following training with the Conjugate-gradient descent algorithm. The row labelled ‘ID3’ shows the performance following training with the ID3 algorithm. Finally, the row labelled ‘NN’ shows the performance following training with the nearest-neighbours algorithm.

The column labelled ‘avoidance-CF’ shows the relative frequency with which the simulated robot crashed into an obstacle while attempting to reproduce the ‘avoidance’ behaviour. The column labelled ‘pursuit-AD’ shows the average (proportional) distance between the simulated robot (producing the pursuit behaviour) and the target. The columns labelled ‘foraging-DF’ and ‘foraging-

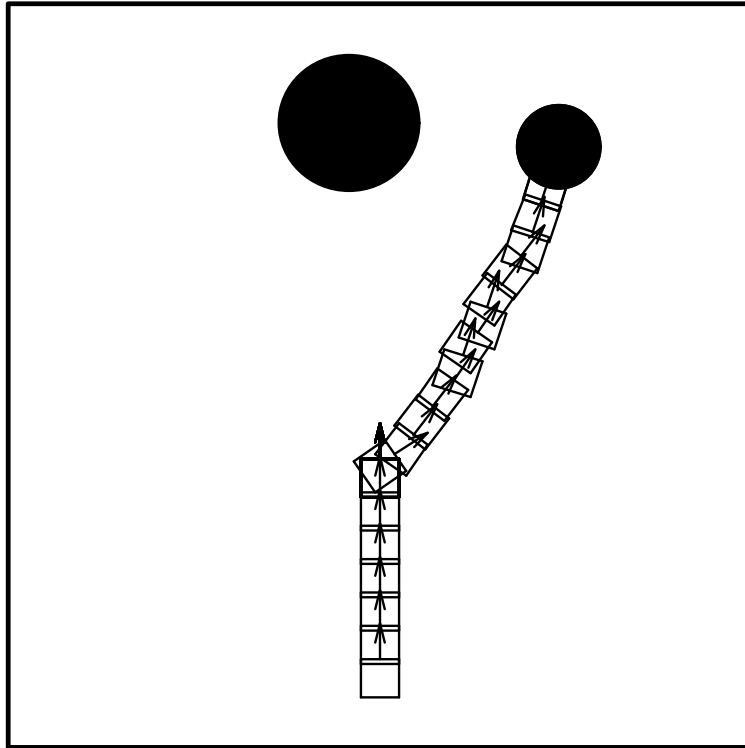


Figure 4:

MF' show the relative frequencies with which the simulated robots (producing the foraging behaviour) moved into a large and a small object respectively. 'DF' here stands for 'Death Frequency'. 'MF' stands for 'Meal Frequency'. All the results are based on training sets of 100 pairs, in which there was equal representation of all possible output cases (i.e., robot motions).<sup>5</sup> Looking at the 'avoidance-CF' column we see that the relative crash frequencies produced by training are fairly low in all cases. This tells us that all the learning algorithms were able to recapture the original behaviour with a reasonably level of accuracy. The situation is the same with 'pursuit'. The low average-distance values show that all the learners performed well on this behaviour. (The robot driven by the ID3 output actually outperformed the hand-simulation.) However, when we look at the 'foraging' columns we see that the reproduced behaviours tended to produce 'Death frequencies' (crashes into a large object) that are equal to and

---

<sup>5</sup>With such a simple simulated world it is essential not to use too large a training set since this may result in the training set including every possible stimulus-response association; i.e., a complete lookup table for the behaviour. In this case, algorithms such as ID3, which build lookup tables in the limit, produce perfect performance necessarily.

	avoidance-CF	pursuit-AD	foraging-DF	foraging-MF
hand-sim	0.000	0.048	0.000	0.778
random	0.078	0.046	0.083	0.000
conjgrad	0.006	0.076	1.000	0.889
ID3	0.006	0.041	0.933	1.000
NN	0.002	0.081	0.917	0.923

Table 2:

in some cases greater than the ‘Meal frequencies’ (crashes into a small object). This tells us that none of the learning algorithms was able to recapture this behaviour.

## 6 Discussion

The results seem to suggest that ‘avoidance’ and ‘pursuit’ are much easier to learn than ‘foraging’. Why should this be? To answer the question we have to consider the statistical regularities of the input-output mappings underlying the relevant training sets. Statistical regularities manifest themselves in three main forms. First we have regularities that are to do with single variables treated independently (the ‘univariate’ regularities.) Second we have the regularities that are to do with relationships between variables (the ‘multivariate’ regularities). Finally, we have regularities that are to do with relationships between *values* of variables, i.e., relationships whose roles are not fixed to particular variables. Let us label these three forms type 1, type 2 and type 3 respectively.

The following training set provides an example of a type 1 regularity. Each line shows a specific training pair. The input appears on the left of the arrow. The output appears on the right. In all cases the inputs and outputs are just bit vectors.

```

1 1 1 1 0 --> 1
1 1 0 0 1 --> 0
0 1 1 0 1 --> 1
1 1 0 1 0 --> 0
1 1 1 0 0 --> 1
1 0 0 0 0 --> 0
1 0 1 1 1 --> 1
1 1 0 1 1 --> 0
0 0 1 0 1 --> 1
1 0 0 1 0 --> 0

```

The regularity here (i.e. the input/output ‘rule’) is simply that the output bit is identical to the third bit of the input vector. This forms a type-1 (univariate) regularity since the effect in question derives from a single input variable treated *independently*.

In the following training set we see an example of a type 2 regularity. The rule here is that the output bit is 1 just in case input bits 2 and 4 satisfy the exclusive-or logic function. This forms a type 2 regularity since the effect derives from a *relationship* between two variables. It is impossible to work out the correct output without taking into account both relevant input variables.

```

0 0 1 1 1 --> 1
0 1 0 1 1 --> 0
1 0 0 1 0 --> 1
0 0 0 0 0 --> 0
0 1 0 0 1 --> 1
1 0 0 0 1 --> 0
1 0 0 1 1 --> 1
1 1 1 1 0 --> 0
0 0 0 1 0 --> 1
0 1 1 1 1 --> 0
1 1 1 0 1 --> 1
1 1 0 1 0 --> 0
0 0 0 1 1 --> 1
1 0 0 0 0 --> 0

```

Finally, the training set below illustrates a type 3 regularity. Here the rule is that the output is 1 just in case there are duplicate values in the input vector. To establish the correct output it is essential to discover whether there are two variables in the input vector that satisfy the equality relationship. However, the siting of the *roles* of the relationship varies from pair to pair. This is an extreme form of multivariate regularity.

```

0.0 0.87 0.62 0.12 --> 0.0
1.0 0.37 0.25 0.75 --> 0.0
0.0 0.0 0.25 0.62 --> 1.0
0.75 0.87 0.62 0.62 --> 1.0
0.0 0.87 0.37 0.12 --> 0.0
0.5 1.0 1.0 0.5 --> 1.0
0.5 0.12 0.0 1.0 --> 0.0
0.0 1.0 0.12 0.87 --> 0.0
0.87 0.12 0.5 0.37 --> 0.0
0.75 0.25 0.12 0.62 --> 0.0
0.12 0.62 0.12 0.12 --> 1.0
0.12 0.5 0.62 0.37 --> 0.0

```

Higher type numbers correspond to more obscure forms of regularity. This suggests that the performance of regularity-exploiting learning algorithms will deteriorate as the type number of the relevant regularity increases. And this seems to be what occurs. All learning algorithms carry out a (possibly implicit) ‘search’ through a space of representations. The goal of the search is to find a representation that accurately reproduces the target input/output mapping.

Since the ‘search spaces’ for some task/learner combinations can be extremely large, it is normally essential to equip the learner with a strong and correct ‘bias’ [9] that will tend to move it in the right direction.

However, even where biases are used, the size of the ‘unbiased’ search space still provides a good measure of the difficulty of the learning task. And when we come to examine search space sizes for learning tasks involving type 1, 2 and 3 regularities we quickly see that tasks involving type 1 regularities are much easier to learn than tasks involving type 2 or type 3 regularities.

In attempting to find a type 1 regularity it is only necessary to work through the relevant variables checking each one for any relevant ‘effects’. When we come to type 2 regularities we have to check all known relationships against all possible n-tuples of variables.<sup>6</sup> To find a type 3 regularity we must check all known relationships against all possible n-tuples of variables in all possible *permutations* of the relevant input-variable sequence. The complexity of both the latter forms of search is extreme.

By applying these statistical observations to the results of the empirical study we can explain the apparent difficulty of ‘foraging’. Foraging is hard to learn precisely because the regularity in the corresponding input/output mapping is type 3. Avoidance and pursuit on the other hand produce training sets which exhibit strong type 1 regularities. Let us see why this is the case.

Consider ‘avoidance’. In this behaviour, particular range inputs treated quite independently have significance within the production of the desired behaviour. If the stimuli (input) associated with a particular range probe is high this denotes an obstacle in close proximity. The desired behaviour is simply to turn away from the relevant direction. This 1-to-1 correspondence between single-variable effects and behavioural responses means that the training set exhibits strong type 1 regularities. Similar remarks apply to the case of ‘pursuit’, except of course that the desired behaviour is now produced by turning *towards* the direction from which there has been a high (i.e., close) range stimulus.

The situation with ‘foraging’ is subtly different however. The reception of a stimulus showing the close proximity of an obstacle has no clear significance within the desired behaviour. The stimulus might have been generated by a large object (meaning the robot should turn away) or by a small object (mean the robot should turn towards the object). To establish the correct response the robot must somehow decide whether there is a *pattern* of stimuli corresponding to a large object or a small object. This entails deciding whether a number of stimuli fit together within a framework of mutual relationships. Given the way the robot moves through the environment, the pattern in question will effectively ‘float’ over the range measures. The roles of the relationships underlying the pattern, then, will not be fixed to particular input variables. The regularity is therefore type 3. From the statistical point of view, then, we would expect the foraging behaviour to be very hard to learn. The empirical study appears to confirm that it is.

---

<sup>6</sup>The value of n here is simply the number of roles in the relevant relationship.

## 7 Summary

The paper has presented an empirical study of three simple behaviours. The study looked at how well ordinary supervised learning algorithms perform when used to train simulated robots to perform certain behaviours. Marked differences in performance were noted and these were explained by drawing a distinction between the three basic forms of statistical regularity.

## References

- [1] Barto, A., Sutton, R. and Anderson, C. (1988). Neuronlike adaptive elements that can solve difficult learning control problems. *Neurocomputing*. MIT Press.
- [2] Cliff, D., Husbands, P. and Harvey, I. (1993). Evolving visually guided robots. In J. Meyer, H. Roitblat and S. Wilson (Eds.), *From Animals to Animats: Proceedings of the Second International Conference on Simulation of Adaptive Behaviour* (SAB92). MIT/Bradford Books.
- [3] Quinlan, J. (1986). Induction of decision trees. *Machine Learning*, 1 (pp. 81-106).
- [4] Rumelhart, D., Hinton, G. and Williams, R. (1985). Learning internal representations by error propagation. ICIS report, Institute for Cognitive Science, UCSD.
- [5] Rumelhart, D., Hinton, G. and Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323 (pp. 533-6).
- [6] Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*. New York: Wiley.
- [7] Johansson, ?, Dowla, F. and Goodman, ?. (1990). Backpropagation learning for multi-layer feed-forward neural networks using the conjugate gradient method. UCRL-JC-104850, Lawrence Livermore National Laboratory.
- [8] Fahlman, S. and Lebiere, C. (1990). The cascade-correlation learning architecture. In D.S. Touretzky (Ed.), *Advances in Neural Information Processing Systems 2* (pp. 524-532.). Morgan Kaufmann Publishers, Los Altos CA.
- [9] Utgoff, P. (1986). Shift of bias for inductive concept learning. In R. Michalski, J. Carbonell and T. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach: Vol II* (pp. 107-148). Los Altos: Morgan Kaufmann.