Innate versus Learned Knowledge

Bridget E. Hallam
Division of Informatics, University of Edinburgh, UK <bridget@dai.ed.ac.uk>

Abstract

Halperin's Neuro-Connector model of learning and motivation requires the inclusion of considerable designer knowledge, but also learns continuously, editing and correcting some of this information. This paper describes some experiments with a computer implementation of this biological model which indicate that fast learning of new material is easily acquired as promised. However, difficulties in unlearning mean that certain problems are expected.

1. Introduction

When a robot is commissioned, it is normally given a task which humans have thought about and for which humans have many ideas about how to achieve good performance. It is sensible to use this expert information in the initial design and programming of the robot. Traditionally this has been the case – robots were entirely pre-programmed by an expert who imagined or did preliminary experiments to decide when and how to switch the robot from one action/behaviour to another.

However, even an expert's ideas may be inaccurate or even plain wrong, for a variety of reasons. It may be reasonable to get the robot to work well for the first day or so — but then an unforeseen problem occurs and what worked in the trials does not work *in situ*. This is particularly true for mobile robots working in unstructured environments, where a change in the environment or in the robot may make part of the robot task require handling in a different way.

If robots are to survive for any length of time in a real, changing, environment, they have to be able to learn whether their behaviour still works, whether it still has the desired effect or not. But researchers in machine learning have often ignored innate knowledge, initiallising connections to around zero or, perhaps worse, random values. The only innate knowledge left has been that included almost accidentally through the way that the system was set up, or through the hardware chosen. It would seem sensible to be able to give machines useful (and editable) basic knowledge and have them add to that *in situ* information relevant to the particular environment in which they are expected to operate. But how?

Animals are born with a considerable amount of innate knowledge about how to survive, and most add to this through learning as they grow and develop. The relative balance between innate and learned knowledge varies considerably: predators appear to use learning where prey animals rely more on innate knowledge, and some insects appear to be almost completely hard-wired.

Therefore, biological models of vertebrate learning have to account for both innate knowledge and life-time learning. These models, where specified reasonably precisely, can be implemented as methods of machine learning. Such implementations are expected to result in agents which can be programmed with whatever knowledge is available, while still allowing this knowledge to be upgraded and refined continuously in the light of the agent's experience. This is exactly what we want for our robots.

Halperin's Neuro-Connector model [1990] is one such biological model. This model was designed to account for the effect of isolation on the aggressive display of Siamese fighting fish, and claimed [Halperin, 1995, p493] as a more general model of processes underlying vertebrate learning. It is reasonably well specified, has been implemented on a robot at a naive level [Hallam et al., 1993], [Saksida, 1994] and looked promising as a means of implementing learning in machines.

A more accurate implementation has been shown [Hallam, 2001] to successfully reproduce the main features of the Neuro-Connector model. Synapse weights increase when delayed S firing is correlated with R firing and decrease when firing is not so correlated. Fast learning and fast unlearning are possible. The model is resistant to forgetting. Behaviours are persistent and can successfully be prioritised, chained, and made self-reinforcing. Behavioural failures can cause weakening of a whole behavioural chain so that chains which are rarely successful can unravel until they are no longer started.

A series of more interesting experiments examining the Neuro-Connector's ability to reproduce various classical conditioning phenomena are reported in Hallam [2001]. Basic classical conditioning and some partial reinforcement and pre-exposure effects have been described in Hallam [2000]. A paper on timing effects is also in preparation.

Theoretical analysis of the model [Hallam, 2001] indicated that learning of new releasers should be fast and accurate as desired, but that there may be problems with the unlearning of inappropriate or less accurate releasing stimuli. Both new learning and correct unlearning are necessary for the development of reliable behaviour in the real world. Investigation of this problem using the accurate computer implementation mentioned above forms the basis of this paper.

2. Halperin's Neuro-Connector Model

Halperin's Neuro-Connector Model [1990] in its simple form is a non-standard neural net in three layers. Input (S) layer neurons are multiply connected to middle (R) layer neurons through adjustable synapses. Middle and output (B) layer neurons are connected on a one-to-one basis providing a feedback loop. The output layer operates a 'winner take all' competition like a competitive net. All other synapses are excitatory.

Input to the system is expected to be some form of sensory data; output is a command to run a particular behaviour from a pre-set repertoire. No direct reverse connection exists between output and input but, since net output is some behaviour in the real world, this is expected to affect sensory input and thus firing in S neurons.

Neuro-Connector components are much more complex than those of standard neural nets, individually incorporating temporal information. The internal dynamics in both neuron and synapse means that the net has an internal time reference independent of the cycle time or number. However, neural output is binary.

Learning in a Neuro-Connector net happens continuously, not just in a preliminary phase. The intuition behind the learning rules is that a successful behaviour will operate for a standard time (different for each behaviour), and that unsuccessful behaviours will get tried for 'too long'. Learning through the environment in this way can only happen for behaviours which actually occur; any others do not get a chance to change the environment and have their performance assessed. For these unexpressed behaviours with unassessible success, intuitively no learning should occur (and no forgetting either). However in fact, as we shall see later, behaviours which are eligible and available but lose the 'winner-take-all' competition for expression are significantly affected by this failure.

Innate knowledge is required by the net both through suitable sensory pre-processing and a suitable pre-existing behavioural repertoire. This avoids many hard problems of perception and actuation but leaves interesting research questions about how the two should be connected.

More importantly, synapses need correct and careful initialisation. This is in direct contrast to many connectionist models where synapses are initialised to random or to identical values. Specific synapse weights need to be set high so that 'good' behaviours (*i.e.*, optimal or nearly so for this stimulus) often occur in the right places. These high-weight synapses can be considered to represent the innate response of many animals to specific 'sign' stimuli.

Neuro-Connector net initiallisation also requires correct setting of an 'expected time' parameter in each synapse. This expected time t_{exp} is related to the time that the behaviour takes to operate under normal circumstances, and is most usefully measured *in situ* although it can be estimated. This provides a value against which errors can be measured. The t_{exp} for each synapse needs to be reasonably precise; synapse weights can be much more coarsely graded.

When the net is in operation, the observed time t_{obs} taken by any behaviour is recorded. Comparing expected and observed times gives a measure of the success or otherwise of the behaviour and therefore gives information as to how synaptic weight should be changed. In essense, if the times match to within a certain 'strengthening window' the synapse is strengthened. If the current behaviour overruns by a little the synapse weakens, though this weakening fades with increased overrun.

In the mathematical implementation of the model [Halperin, 1990, appendix I] the opposite side of the strengthening window (where the behaviour is too short) is marked by strong weakening. In the original, textual description of the model any small, positive, t_{obs} is defined as being strengthening. The textual description is considered the more authoritative. Small, positive, t_{obs} occur frequently in nets of reasonable size — any time that an R fires but its B neuron loses the winner-take-all competition. Textual and mathematical descriptions can be reconciled if t_{exp} is limited to less than the width of the strengthening window. This limitation on t_{exp} is unrealistic and not always adhered to in the experiments following, for reasons given in the text. The mathematics given for the model could do with being more precisely crafted. This was attempted in Wilson [2000].

This automatic assessment of the net's 'decision making' ability through the timing of any effect on the environment makes a Neuro-Connector net self-supervising. A Neuro-Connector net starting with good innate knowledge is expected to quickly learn the correct (*i.e.*, most likely to be successful) behaviour to perform under each distinguishable sensory state. It cannot learn expected times.

In the original model, a single stimulus causes firing in a set of S neurons distributed in an S-neuron pool which overlaps with similar pools. R and B neurons exist in non-overlapping pools. In the experiments following the situation has been simplified so that one simulation neuron represents the whole set of neurons responding to a single stimulus. This simplification qualitatively affects the R/B feedback loop but is not thought to significantly affect the results given here.

In this simulation, the action of the behaviours is to run code to change S neuron input. When used on a robot each behaviour will not affect S input directly as here but will do some activity expected to produce a change in stimulus state. It is this change in stimulus state which will be detected by the Neuro-Connector net in the form of new Ss firing. Thus successful behaviour in the environment will be necessary for the production and maintenance of behavioural chains.

3. Basic Simulation Features

The simulation was checked to ensure that it gave synaptic weight changes that agreed with weight change rules specified by the model. The results, given in figure 1, clearly show how the relationship between t_{exp} and t_{obs} affect synaptic weights. Correct synapse weight changes are the main verification needed for my implementation of a Neuro-Connector net, but are only half the story. For any real application we need behaviours to connect, forming chains of activity that start with one particular sensory state and end when the task is done. A side-effect of prioritisation, discovered during preliminary testing, is also discussed below.

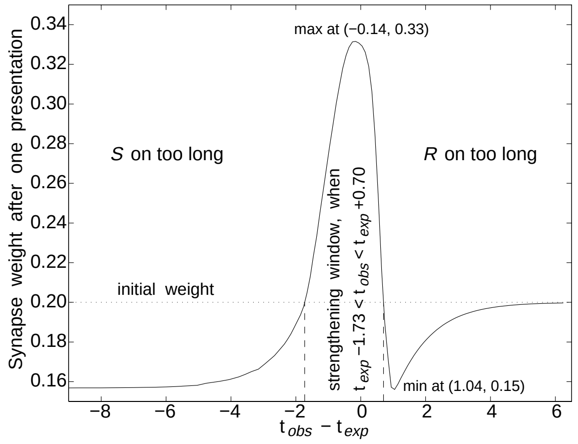


Figure 1: Weight Changes at Different $t_{obs} - t_{exp}$.

3.1 Fast Learning

When S and R fire in a perfect time relationship, synaptic weight increases from the minimum of 0.01 to over the R threshold of 0.4 in five presentations. With less perfect relationships, more presentations are needed.

3.2 Resistance to Forgetting

When neither S nor R fire, no weight change takes place. When R fires alone, no weight change takes place. However, resistance to forgetting implies that no weight is lost even when the stimulus is seen, if there are more important events happening. Under what circumstances can the stimulus be seen and synapse weight not be lost?

Obviously, if R and B fire in a correct time relationship with the S, strengthening occurs. However, then the stimulus presentation is reinforced by the behaviour so forgetting would not be expected. If B does not fire so R is not held on, then R firing finishes with S end-of-firing giving a t_{obs} of around 0.1. This is defined as strengthening in the model but is not supported by the mathematical equations given to instantiate the model. Therefore resistance to forgetting of this form happens in the model but not in the implementation unless t_{exp} is under 1.7. In the model, high-weight synapses (where R firing always follows S) can only decrease in weight when the behaviour happens but overruns. In other words, forgetting can only happen if the behaviour occurs.

This inability to forget the connection may become a problem, since the net may gradually fill with high-weight synapses which cannot be forgotten. Too many high-weight synapses would be expected to increase total R and B input and therefore encourage the production of high-priority behaviours at the expense of low-priority behaviours.

With neural pools, this 'no forgetting' problem may be attenuated. Where several inputs are needed for R neurons to fire, none is individually above R threshold. Therefore it is possible that R firing will not 'follow' S firing making it possible for even maximum-weight synapses to decrease, but only if S firing is totally unsupported by firing in other Ss connected to the same R.

3.3 Prioritising Behaviours

It is important that behaviours can be prioritised easily, both to allow emergency over-ride of the current behaviour and to allow behavioural chains to be crafted. Both situations require that it is possible to notice when behaviours first have input over threshold so that if any of them have a higher priority than the behaviour currently active the new behaviour can interrupt.

This prioritising causes important behaviours to be performed as soon as appropriate stimulus conditions occur. Lower-priority behaviours are interrupted and/or have to wait. This may mean that the sensory neurons releasing the low-priority behaviours are reaching the end of their adaptation time by the time that the high-priority behaviour has finished and the low-priority behaviour can emerge. Then S firing will finish too early, causing t_{obs} to be too long and causing weakening which may well not be appropriate. This effect can be shown in simulation but is presumably not likely to occur often in practise.

In other words, the necessary prioritising of behaviours means that high-priority behaviours and weight changes associated with them will operate correctly but weight changes associated with lower-priority behaviours may not be reliable. Under these conditions, fast weight changes may be undesirable.

With the large neurons pools of the full model the choice of behaviour could depend upon the number of R neurons providing input to the B pool as well as the priority of the behaviour, making this problem less severe.

3.4 Persistence

It is essential that important behaviours should be able to persist despite changes in the environment. An animal doesn't want to be distracted from fleeing a predator even if it runs past some unexpected food. The Neuro-Connector model models persistence extremely well due to the prioritising of B neurons meaning that any behaviour will persist until either it has finished or else a higher-priority behaviour interrupts it. If we assume that emergency actions have high priority then persistence is not a problem for these. The interesting thing is what happens to synapse weights connected to the stimuli which are being ignored due to the emergency.

Since lower-priority behaviours are not being expressed (their Bs try to fire, lose the B competition, and so stop), their R neurons fire with exactly the same timings as their high-weight-connected S neurons. t_{obs} is therefore 0.1 for all synapses whose weight is above R threshold. The model specifies that this is strengthening but the mathematics given does not support this unless t_{exp} is between zero and 1.7. If $t_{obs} = 0.1$ is a strengthening time relationship then synapse weight will increase on successive presentations of (ignored) S until it reaches maximum. If t_{exp} is bigger than 1.7 then the weight will decrease until it goes below R threshold, *i.e.*, the association is lost or forgotten. At that point R will not fire (or, at least, will not be made to fire by these Ss). S on without R is defined as causing significant weakening. These synapses will therefore have their weight reduced quickly each time the stimulus is seen until the weight reaches the minimum allowed. Thus all weights are expected to be near to either maximum or minimum nearly all the time. This will seriously interfere with the learning of the correct releasing stimuli for low-priority behaviours since weights will not stay at intermediate levels if more significant stimuli are seen alongside those for the low-priority behaviour.

Unfortunately, if $t_{obs} = 0.1$ is not strengthening, the low-priority behaviour is then lost, since the 'best' releasing stimuli for that behaviour were firing correctly and their synapses were weakened anyway. This means that, unless the loss of low-priority behaviours does not pose a problem, either $t_{obs} = 0.1$ *must* be strengthening or else at least one 'correct releasing stimulus' for each behaviour must have its synapse to that R/B pair fixed at a high enough weight. Since one of the stated strengths of this model is that releasing stimuli do not need to be known *a priori*, preliminary testing is implied during which low priority behaviour expression is protected, so that the appropriate stimuli can be found. Another solution would be reinvention of the mathematical equations governing weight changes. Changing the mathematics of this model in this way is more complex that it looks initially, due to cyclic dependencies between various parameters.

3.5 Self-reinforcing Behaviours

Chains of behaviour need to be reinforced by the successful completion of the next behaviour in the chain, and so ultimately by the success of the final behaviour. The 'inside' behaviours are expected to be followed by the next behaviour in the chain in the correct sequence and at the correct time, but there is no expectation of which behaviour will follow a chain-terminator. So this unknown next behaviour cannot be expected to follow at any particular time interval. Since t_{exp} cannot be specified, chain-terminating behaviours need to be self-reinforcing. These behaviours act as rewards, confirming whatever sequence of behaviours was previously expressed, and are guaranteed to strengthen synapses between their R neurons and the appropriate releasing stimuli. How can this guarantee be achieved? Some alternatives are indicated below.

1. Include code in the behaviour to turn off its S neuron at the correct time (*i.e.*, t_{exp} before it itself adapts out. This operates correctly if that behaviour isn't interrupted before the S is switched off and if the S hasn't already adapted out. If the behaviour has a high enough priority then these are no problem as it will not be interruptible, and since it will start as soon as its S starts firing its S will not have time to adapt out unexpectedly. Making self-reinforcing B neurons adapt earlier than most may also help, since this reduces the amount of time available for interruptions to take place. This will not work if other Ss hold the R on.

2. Set S and B adaptation times so that the neurons will finish firing t_{exp} apart, thus forcing

$R - S$ to be within the strengthening window. This may fail if other Ss hold R on, and assumes that this behaviour can immediately interrupt whatever behaviour was currently active when this S started firing.

3. Fix the synapse weight. This would ensure that these synapses did not weaken under any circumstances, but would prevent the unlearning of inaccurate releasers (see section 6.).

Perhaps a more significant problem with all the above approaches is that they require the appropriate 'releasing' stimulus to be known *a priori*. The 'correct' stimulus is known by the program in all experiments performed in this paper, but is more problematic in real life. A programmer would have to make a guess which would work under most circumstances, and use preliminary testing to refine this guess and discover the best releasing stimuli for each behaviour. If preliminary testing is not possible it may be necessary for the end behaviour of a chain to turn off (point 1 above) or adapt in a correct time relationship with (point 2) all the S neurons in its group. The grouping of neurons into sets is a feature of most ethological models, so this may not be a conceptual problem, but would mean that all these S neurons would develop and maintain high-weight releasing synapses with the self-reinforcing behaviour's R pool, resulting in this behaviour being produced rather (too?) often.

If a t_{obs} of shorter than t_{exp} is always strengthening, as specified by the textual description of the model, then another possibility arises. Self-reinforcing behaviours could then have synapses with long t_{exp} so that almost any time relationship was strengthening. The advantage of this approach is that it does not require *a priori* knowledge of releasing stimuli nor preliminary testing. The disadvantage is that any synapse with a long t_{exp} will get strengthened under more circumstances than synapses with shorter t_{exp}. My implementation cannot use this method because it implements the mathematical version of the model and not the textual version. This may or may not be a disadvantage!

Note that self-reinforcing behaviours must be used with care, as every time that they occur many synapses will be strengthened: they cannot, by definition, occur with incorrectly long timings.

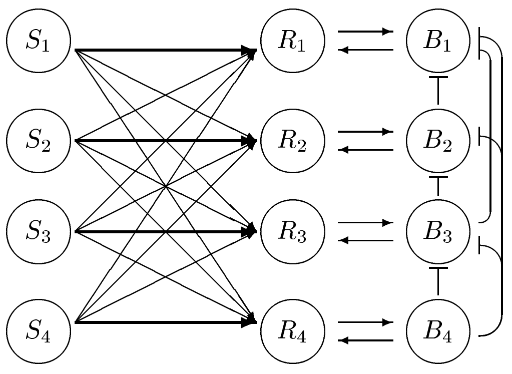
3.6 Behavioural Chaining

Most visible behaviour is composed of sequences of smaller units, some of which may be relatively invariant. These invariant sequences can be performed without much thought or external prompting. They can be thought of as 'habits', or as 'routines' [Agre and Chapman, 1991] and form the basis of more complex behaviours.

Chaining behaviours implies that the Bs are prioritised and can be called in order, so that each B can interrupt the previous one as soon as the correct sensory situation occurs. In my implementation the ordering is achieved by making each behaviour switch on an S which releases the next-highest-priority behaviour. Imagine that it is some aspect of the successful completion of the prior behaviour that creates the sensory situation that triggers the subsequent behaviour. Strictly speaking a successful chain does not *require* differential prioritisation of behaviours since each B can switch on an S just before its own adaptation time removes it from the behaviour competition. But the B adaptation time needs to be set longer than this so that we are able to detect unsuccessful behaviours by their overrunning. Prioritising therefore becomes important when we need to be able to decide whether a behaviour is successful or not.

3.6.1 Chaining Experiment

A Neuro-Connector net comprising four S, four R and four B neurons was set up as in figure 2, and high-weight connections (0.6) made between S and R pairs with the same index number. t_{exp} was set at 8 time units for all synapses. Three behaviours were designed that each waited for 6 time units, switched off their own releasing stimulus, waited another 8 time units, and switched on the releaser for the next behaviour. B adaptation time was 17 and their refractory time 50. The fourth behaviour was self-reinforcing, switching off its own stimulus at time 8.5, roughly t_{exp} before its own adaptation time caused it to time out.



B_1 sets S_1_input = 0 and S_2_input = 1
B_2 sets S_2_input = 0 and S_3_input = 1
B_3 sets S_3_input = 0 and S_4_input = 1
B_4 sets S_4_input = 0 just before it times out

Figure 2: Net Used for the Chaining Experiments.

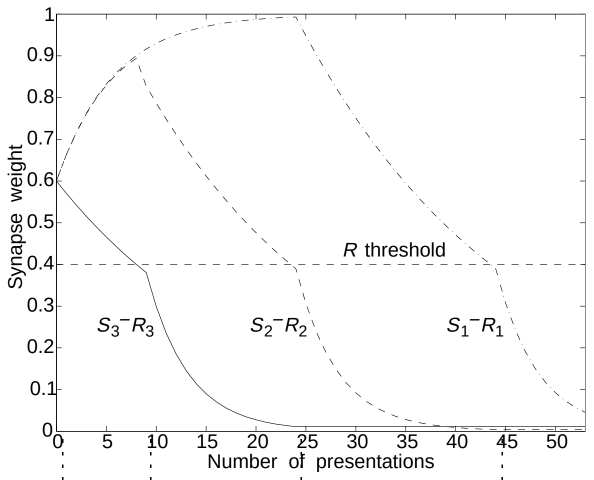
Following presentation of stimulus 1, all behaviours were produced in order, as desired. After 25 such reinforced presentations all high-weight connections had increased weight from 0.6 to 0.99. All low-weight connections stayed low.

This experiment shows that the net can produce behavioural chains such as proposed for the example application given in Halperin [1991] of a robot fly detecting and labeling plastic litter.

3.7 Backward Chaining of Failure

Halperin [1991], in describing the theoretical behaviour of her model, specifies that a robot performing a sequence of actions which is consistently unsuccessful will learn not to start the sequence. This was tested in simulation as described below.

The last experiment was repeated but with the penultimate behaviour 'failing' by not switching on the releasing S for the final behaviour. Since the final behaviour did not occur, the penultimate behaviour overran. This weakened its releasing $S \rightarrow R$ synapse until eventually this behaviour was not called and the previous behaviour overran. You may find it helpful to imagine the agent continuing to try the behaviour without success, and so without changing the sensory state in the required manner. Each behaviour in turn overran, keeping its R on too long and weakening the releasing $S \rightarrow R$ connection, dropping out in turn as illustrated in figure 3.



Observed Behaviours:

not B_4	B_1 and B_2	B_1 only

Figure 3: Synaptic Weights and Observable Behaviours Show the Backward Chaining Effect of Unsuccessful Behaviours.

The reason for each behaviour taking longer to drop out than the previous one (later in the chain) is that early behaviours continue being reinforced by the correct finishing of later behaviours until the behaviour just after them in the chain fails. Therefore synapse weight still increases for early behaviours even while the final behaviours are failing (figure 3b). All releasing synapse weights started at 0.6; the last surviving high-weight synapse had reached 0.993 before its weight started falling. After 44 presentations none of the behaviours started. This shows that the agent can learn not to start a whole sequence of behaviours which is known not to be successful. This supports the claim of Halperin [1991] that a robot fly with the task of labelling plastic could learn not to waste time on plastics that were not labellable.

Since there were no other releasing stimuli for any of these behaviours in this experiment, these behaviours were lost. In a more complex system other releasing stimuli will be present. As a minimum, there will be one releasing stimulus for each of the slightly different circumstances in which these behaviours should operate. There will be overlap in the sensory neuron representation of these stimuli, but hopefully not so much that 'correct' and 'incorrect' releasing stimuli cannot be distinguished. Any Ss which have not fired in these particular 'bad' circumstances will not have lost their releasing ability.

This experiment proved very difficult to initialise correctly because the weakening that happens when R is on 'too long' fades out so fast with increasing overrun. Initially B adaptation time was set at 20 time units and weakening was too slow to be practical. Even with B adaptation time set at 18 time units instead of the 17 finally used it took 32 presentations for the penultimate behaviour in the chain (the first to disappear) to cease to appear.

This difficulty implies that synaptic weakening caused by R being on 'too long' is likely to be slow, since the timing required to produce maximal weakening has to be so precise.

4. Learning New Releasers

This section tests the claim that a stimulus which accurately predicts when a behaviour should be performed should become a new releaser for that behaviour if the behaviour is normally performed correctly.

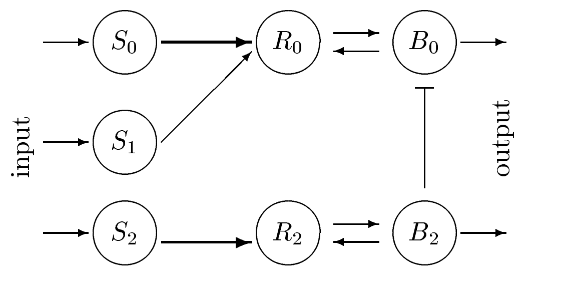


Figure 4: Simple Net Used for Training a New Releaser.

A Neuro-Connector net was set up with three S neurons and two R/B pairs, as illustrated in figure 4. $S_0 \rightarrow R_0$ weight started at 0.6 so that S_0

firing was always followed by R_0 and B_0. $S_1 \to R_0$ weight started at 0.01. A cycle of five presentations was given:

1. S_0 and $S_1 \to$ correct behaviour;
2. S_0 alone $\to$ incorrect behaviour, B_0 overran;
3. S_0 and $S_1 \to$ correct behaviour;
4. S_1 alone, leading to correct behaviour once the association had been learned; and
5. S_0 and $S_1 \to$ correct behaviour.

One out of every four presentations of S_0 failed in that the behaviour was made to overrun, to simulate the behaviour being called incorrectly and being unsuccessful. S_1 firing correlated with S_0 firing when the behaviour followed correctly but not when the behaviour overran. S_1 also fired without S_0 on every fourth presentation of S_1, to simulate circumstances where the behaviour was required but was not produced by default. S_2 was used to call B_2, which then interrupted B_0 at the correct time (or not, when S_0 was seen alone). The idea behind this is that correct operation of B_0 would, in real life, naturally result in a change in the environment equivalent to stimulus 2 being seen. Incorrect operation of B_0 does not result in this particular environmental change, although it may result in another change which does not affect firing in B_0.

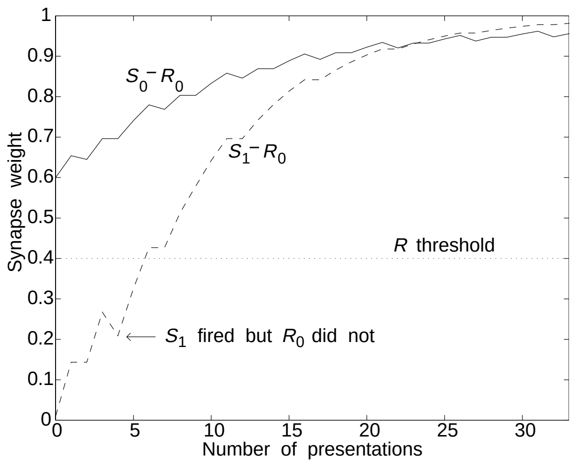


Figure 5: Stimulus Refinement: Learning a New Releaser.

Figure 5 shows the weights obtained for synapses $S_1 \to R_0$ and $S_0 \to R_0$. The decrease in $S_1 \to R_0$ weight when S_1 fired alone at presentation 4 can clearly be seen, as can the flat sections in both lines where the synapse did not change during one presentation because the S did not fire. The decrease in $S_0 \to R_0$ weight at presentations 2, 7, 12 ...when S_0 appeared alone and the behaviour failed is less obvious since the behaviour overran by 4 time units which is too long to allow much weakening.

These results show that a new releaser is easily learned — in this case in only five presentations.

5. Unlearning Releasers

The above experiment showed that S neurons which are followed correctly by the behaviour 75% of the time can keep being used as releasers of that behaviour. What proportion of failure is needed before the synapses will weaken out? This answer depends upon the ratio of strengthening on the 'correct' trials and weakening on the 'incorrect' trials. Both are very variable, depending upon the precise match between t_{exp} and t_{obs} on both strengthening and weakening trials. Significant weakening only occurs (according to the mathematics developed by Halperin [1990]) if there is only a short behavioural after-discharge so that R overruns only a little. If the behaviour overruns too much then very little weakening occurs on the 'failed' trials so only a very small proportion of 'successful' trials are needed to maintain the releaser.

The precision required in the behavioural overrun suggests that there may be a problem in unlearning releasers as this sort of consistent precision is biologically implausible, as is the alternative that almost no successful trials occur (since releasers are chosen to normally operate correctly).

6. Unlearning Imprecise Releasers

The experiment above showed that a certain amount of inaccuracy in releasing stimuli (or, a certain failure rate in the performance of the behaviour) does not upset synapse weights sufficiently for releasers to be lost. This is considered a good feature of the model, in that we don't want to lose good releasers just because the designer misimagined sensor readings or an emergency or low battery caused anomolous behavioural timings.

However, imprecise releasers, *i.e.*, those stimuli which may appear when different behaviours are appropriate and which have a high-weight connection to one of these behaviours, are a particular problem and should be pruned. There are many circumstances under which initial learning is too general and the 'releasing' stimuli need to be refined until only the 'best', earliest, and most accurate stimulus causes a response. Could such a 'common' S neuron, which stimulates enough R neurons to release more than one behaviour, be successfully demoted?

To test this, a Neuro-Connector net was set up with three S neurons and two R/B pairs as illustrated in figure 6. S_1 responds to stimulus 1 which is the correct releasing stimulus for behaviour 1. S_2 responds to stimulus 2 which is the correct releasing stimulus for behaviour 2.

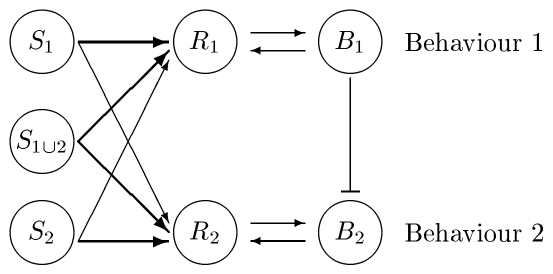


Figure 6: An S which Responds Too Often.

$S_{1\cup 2}$ responds to both or either stimulus. t_{exp} was set at 1.7 for all synapses. Synapse weights for $S_1 \rightarrow R_1$, $S_{1\cup 2} \rightarrow R_1$, $S_{1\cup 2} \rightarrow R_2$, and $S_2 \rightarrow R_2$ started above R threshold, at 0.6. B adaptation time was 7 time units. S neurons were switched off by the behaviours six time units after the behaviour started. Alternate presentations of stimulus 1 and stimulus 2 were given. $S_{1\cup 2}$ fired concurrently with either.

When stimulus 1 appeared, S_1 and $S_{1\cup 2}$ fired, both Rs fired, and behaviour 1 appeared. When stimulus 2 appeared, S_2 and $S_{1\cup 2}$ fired. $S_{1\cup 2}$ stimulated both R_2 and R_1. Behaviour 1 therefore appeared after every presentation because it had highest priority. t_{obs} for all active Ss with R_1 was 0.8. t_{obs} for all active Ss with R_2 was 0.1. Weight changes are shown in figure 7.

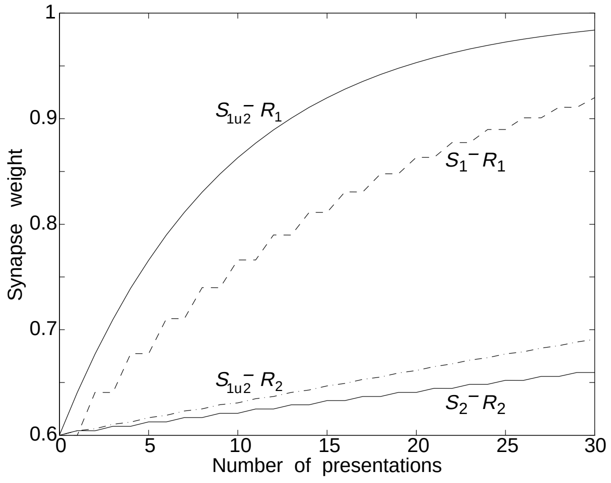


Figure 7: Unlearning Imprecise Releasers: Failure.

All releasing synapses strengthened, as shown in figure 7. $S_{1\cup 2} \rightarrow R_1$ strengthened most because both fired correctly at every presentation. The t_{obs} of 0.8 does not match the t_{exp} of 1.7 very well, but enough to allow strong strengthening. $S_1 \rightarrow R_1$ strengthened similarly when it fired, which was on alternate presentations. When S_1 was not seen, a marginal decrease in weight occurred. $S_{1\cup 2} \rightarrow R_2$ strengthened a little because at every presentation R_2 firing 'followed' that of $S_{1\cup 2}$, giving a small positive t_{obs} of 0.1. According to the model, all small positive t_{obs} are strengthening. This small amount of strengthening occurred on every presentation. $S_2 \rightarrow R_2$ strengthened similarly when S_2 was seen, which occured on alternate presentations. When S_2 was not seen, a marginal decrease in weight occurred. Strong strengthening was also observed in synapse $S_2 \rightarrow R_1$ (not shown) whenever S_2 was seen, because this synapse also had a t_{obs} of 0.8 and a t_{exp} of 1.7. After five presentations of S_2 (experiment presentation 9) this weight had increased from 0.01 to 0.41, above R threshold. By presentation 30, this weight was 0.80. Therefore S_2 became a releaser for behaviour 1 after only five presentations of S_2.

The lower-priority behaviour 2 will never be expressed despite high-weight synapses driving its R neuron, because one of its releasers also releases a higher-priority behaviour. This is expected to be a problem under most circumstances.

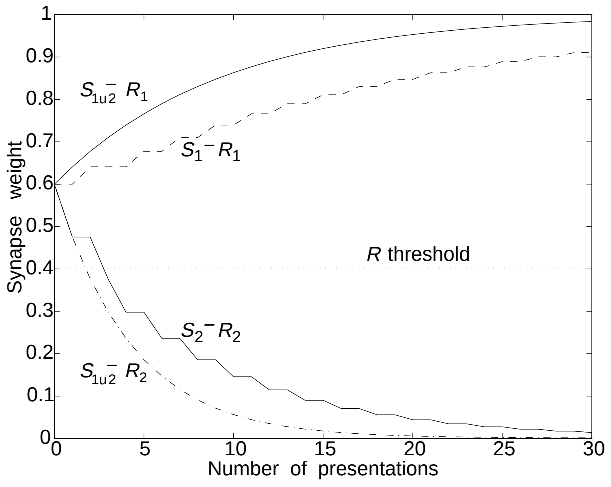


Figure 8: Unlearning Imprecise Releasers: Partial Success.

6.1 *Using a Larger* t_{exp}

The above experiment was repeated with a t_{exp} of 5 on all synapses. B adaptation time was adjusted to 10.3 time units to make this t_{exp} strengthening when the behaviour ran correctly. However, the t_{obs} of 0.1 seen when an R neuron merely 'follows' an S rather than the R being held on by the activity of its B neuron is now weakening.

When either stimulus appeared $S_{1\cup 2}$ stimulated both R_1 and R_2, as before, and behaviour 1 again appeared because it had highest priority. Weight changes can be seen in figure 8.

All synapses with R_1 strengthened as before. This was as expected because the experiment is effectively the same as before for these synapses.

By contrast, synapses with R_2 weakened every time they were active (*i.e.*, the S fired). R_2 'followed' S firing as before, but in this example $t_{obs} = 0.1$ was *not* strengthening because t_{exp} is 5 and the time window where strengthening occurs is only 3 time units wide with the parameters used here.

As before, behaviour 2 is never expressed, but in this experiment active synapses with R_2 weaken until it is not possible to drive that behaviour through those synapses. Any S neuron which fires in response to common stimuli and is involved in releasing a high-priority behaviour will have this effect on other neurons firing alongside it – the highest priority behaviour will be expressed and correctly-firing S and R neurons for lower priority behaviours will get their synapses weakened. In this example, behaviour 2 not only never appears, but also loses all its specific releasers so is completely lost!

Note, however, that this problem only occurs when small positive t_{obs} are weakening. This is specifically *not* in accordance with Halperin's [1990] textual description of the model, where short, positive t_{obs} are assumed to be always strengthening, although it is a direct result of implementing the mathematics given by Halperin [*ibid*, Appendix I].

These results show that weakening of some imprecise synapses can occur if $t_{obs} = 0.1$ is *not* strengthening, but only for synapses with low-priority behaviours. This partial success is not a true solution to the 'imprecise releaser pruning' problem — the synapse from the 'common' S neuron to the highest-priority behaviour still remains high-weight.

7. Not Learning Imprecise Releasers

Obviously many of the above problems can be avoided if there are no imprecise releasers. A programmer should take care not to initialise a net with a high starting weight for an S which responds to general or common stimuli: this is hard but can be done. But do these stimuli naturally become releasers during normal net operation?

To answer this question, the experiment above with t_{exp} of 1.7 was repeated with starting synapse weights of 0.01 from $S_{1\cup 2}$, to prevent this neuron releasing behaviour 1 on initial presentations. In this experiment behaviours 1 and 2 are output alternately. Synaptic weights are shown in figure 9.

When synapses with $S_{1\cup 2}$ start at a low weight then when stimulus 1 appears and behaviour 1 is performed correctly $S_1 \rightarrow R_1$ and $S_{1\cup 2} \rightarrow R_1$ strengthen, $S_{1\cup 2} \rightarrow R_2$ weakens and $S_2 \rightarrow R_2$ remains unchanged. When stimulus 2 appears and behaviour 2 is performed the reverse applies: $S_2 \rightarrow R_2$ and $S_{1\cup 2} \rightarrow R_2$ strengthen, $S_{1\cup 2} \rightarrow R_1$ weaken and $S_1 \rightarrow R_1$ remains virtually unchanged.

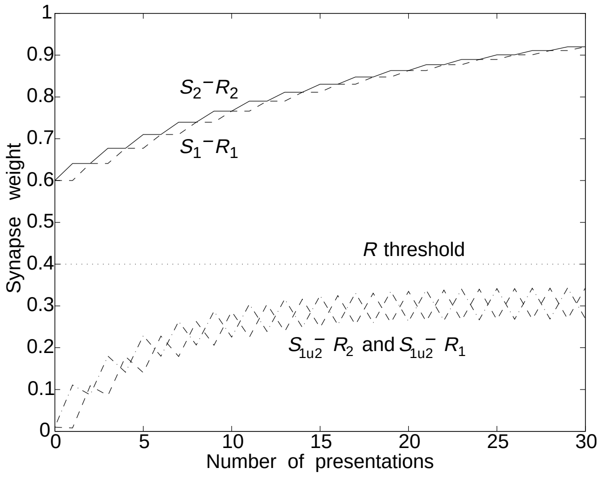


Figure 9: Not Learning Imprecise Releasers.

After 13 alternate successful presentations of each stimulus (experiment presentation 26), $S_1 \rightarrow R_1$ and $S_2 \rightarrow R_2$ reached 0.9. $S_{1\cup 2} \rightarrow R_1$ and $S_{1\cup 2} \rightarrow R_2$ oscillated, increasing when their R fired and decreasing when S went on alone. The overall tendency was negatively accelerating strengthening, towards approximately 0.27 and 0.35 on alternate presentations. The exact weight of synapses with $S_{1\cup 2}$ is rather arbitrary, depending upon the ratio of stimulus 1 to stimulus 2 and the relative importance of strengthening and weakening effects, *i.e.*, the precise match and mismatch between t_{exp} and t_{obs} on strengthening and weakening runs. Alternating presentations kept $S_{1\cup 2} \rightarrow R_1$ and $S_{1\cup 2} \rightarrow R_2$ below R threshold with the parameters chosen. However, these weights went above R threshold immediately if the same stimulus was presented twice in succession with the particular gains and timings used here.

If our 'common' S fires in response to other stimuli also (perhaps it detects a common situation such as 'something within 10cm') then the weakening is expected to predominate, because for any given synapse there will be more presentations of the 'wrong' stimulus, so more opportunities for weakening as other behaviours are chosen, and fewer opportunities for strengthening.

7.1 With Larger t_{exp}

The above experiment was repeated with a t_{exp} of 5 on all synapses, making a t_{obs} of 0.1 weakening as in section 5.1 above. B adaptation time was again adjusted to 10.3 time units to make this t_{exp} strengthening.

The results obtained were essentially the same as those in figure 9: after 13 alternate successful presentations of each stimulus $S_1 \rightarrow R_1$ and $S_2 \rightarrow R_2$ reached 0.9. $S_{1\cup 2} \rightarrow R_1$ weight at presentation 30 was 0.27 and $S_{1\cup 2} \rightarrow R_2$ weight was 0.34. The precise numbers involved are marginally different from those of the last experiment due to the slightly different t_{obs}/t_{exp} relationship in this experiment.

8. Discussion

New accurate and precise releasers can easily be learned, but inaccurate and imprecise releasers are also learned and may be hard to remove again. This means that the total number of synapses with enough weight to make releasing stimuli is expected to increase during the robot's lifetime. This is not desirable as it will cause the loss of low-priority behaviours.

It can clearly be seen that great care is needed when setting up high-weight synapses, since it is hard to lose them. Synapses connecting to S neurons that respond to specific stimuli are greatly to be preferred over Ss which are more general. Ss which fire in response to a common stimulus (*e.g.*, the 'paleness-detecting' Ss of Halperin [1991]) cause problems if they are able to release a behaviour of higher priority than the one appropriate to the specific stimulus situation. Unfortunately these Ss may well develop releasing connections under normal net operation.

The main hope for not strengthening synapses with 'common' Ss is that they respond to so many stimuli that their firing pattern will often not be correlated with the specific stimulus about which we are learning. The common Ss probably started firing too early and are refractory, else stay on too long. Either would interfere with learning.

Another obvious problem noticeable from these results is that 'common' S neurons can block the expression of low-priority behaviours. This may be a problem in itself, and could also result in the lower-priority behaviours vanishing irretrievably if a t_{obs} of near 0.1 is not strengthening. (t_{obs} on high-weight synapses connected to behaviours which are not expressed is always about 0.1.) It may be possible to change Halperin's mathematics to solve these problems without needing to change the textual description of the model.

It is possible that a 'grandmother cell' representation would suffer less from these problems that the distributed representation assumed, although this may be less biologically plausible.

In conclusion, Halperin's Neuro-Connector net can be set up to produce fast learning and unlearning, behavioural chains that can unravel under appropriate circumstances, and quick learning of new releasers. The initialisation required needs to be precise as regards timings, but since these timings can be measured from preliminary experiments, this is not seen as a problem.

The main problems arise from a difficulty in reducing weights without needing implausibly accurate timings. This will result in input neurons which respond to too many stimuli may become too influential in behavioural choice, causing higher-priority behaviours to be executed too often at the expense of lower-priority behaviours. As robots survive for increasing periods in real environments, and especially if these robots are developing themselves in any way, this will become an increasing problem.

The idea of incorporating considerable designer knowledge into a robotic system *and allowing this information to be refined as its inaccuracies are discovered* is excellent and will be necessary for epigenetic robots. However, Halperin's Neuro-Connector model needs more development before it is entirely suitable for this role.

References

P. E. Agre and D. Chapman. What are plans for? In P. Maes, editor, *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, pages 17–34. The MIT Press, 1991.

B. E. Hallam. Simulating classical conditioning using a neuro-connector net. In J-A. Meyer, A. Berthoz, D. Floreano, H. Roitblat, and S. Wilson, editors, *From Animals to Animats 6*, pages 279–286. MIT Press, 2000.

B. E. Hallam. *Simulating Animal Conditioning: Investigating Halperin's Neuro-Connector Model.* PhD dissertation, University of Edinburgh, 2001.

B. E. Hallam, J. R. P. Halperin, and J. C. T. Hallam. An ethological model of learning and motivation for implementation in mobile robots. Research paper 629, University of Edinburgh, 1993.

J. R. P. Halperin. *A Connectionist Neural Network Model of Aggression.* PhD dissertation, Department of Ethology, Toronto University, Canada, 1990.

J. R. P. Halperin. Machine motivation. In J-A. Meyer and S. Wilson, editors, *From Animals to Animats*, pages 213–221. MIT Press, 1991.

J. R. P. Halperin. Cognition and emotion in animals and machines. In H. Roitblat and J-A. Meyer, editors, *Comparative Approaches to Cognitive Science*, chapter 19, pages 465–499. A Bradford Book. MIT Press, Cambridge, MA, 1995.

L. M. Saksida. Instrumental conditioning of a mobile robot: The use of the Halperin Neuro-Connector model for robot control. Master's thesis, Artificial Intelligence Dept, 1994.

P. Wilson. Refining the Neuro-Connector model. Master's thesis, Division of Informatics, UK, 2000.