Available online at www.sciencedirect.com
SCIENCE DIRECT®
Learning and Motivation 36 (2005) 88–103
Learning and Motivation
www.elsevier.com/locate/l&m

The implications of null patterns and output unit activation functions on simulation studies of learning: A case study of patterning

Vanessa Yaremchuk[a], Leanne R. Willson[b], Marcia L. Spetch[c], Michael R.W. Dawson[a,*]

[a] *Biological Computation Project, Department of Psychology, University of Alberta, Edmonton, Alba., Canada T6G 2E9*
[b] *Taylor University College 11525-23 Avenue, Edmonton, Alba., Canada T6J 4T3*
[c] *Department of Psychology, University of Alberta, Edmonton, Alba., Canada T6G 2P9*

Received 21 May 2004; received in revised form 1 October 2004
Available online 8 December 2004

Abstract

Animal learning researchers have argued that one example of a linearly nonseparable problem is negative patterning, and therefore they have used more complicated multilayer networks to study this kind of discriminant learning. However, it is shown in this paper that previous attempts to define negative patterning problems to artificial neural networks have specified the problem in such a way that it is much simpler than intended. The simulations described in this paper correct this problem by adding a "null" pattern to the training sets to make negative patterning problems truly nonseparable, and thus requiring a more complicated network than a perceptron. We show that with the elaborated training set, a hybrid multilayer network that treats reinforced patterns differently than nonreinforced patterns generates results more similar to those observed by Dalamater, Sosa, and Katz in animal experiments than do traditional multilayer networks.

* Corresponding author.
E-mail address: mdawson@ualberta.ca (M.R.W. Dawson).
URL: www.bcp.psych.ualberta.ca/~mike/.

0023-9690/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.lmot.2004.10.001

Keywords: Discrimination learning; Perceptron; Negative patterning; Positive patterning; Connectionism

Introduction

The perceptron, learning, and linear separability

The perceptron is a simple artificial neural network (ANN) that can learn to categorize patterns. In general terms, it consists of a set of input units that are used to represent the values of stimulus variables. For example, one input unit could be associated with conditioned stimulus 1 (CS_1), and would be turned on if CS_1 was present, and turned off if CS_1 was absent. Connections with modifiable weights are used to link the input units to one or more output units. The input units send signals to the output units through these connections. The activation from an input unit is scaled by the connection by being multiplied by the connection weight when the signal is sent. As a result, a connection weight is analogous to the associative strength of a conditioned stimulus.

On the one hand, perceptrons would seem to be of considerable interest to animal learning researchers (e.g., Pearce, 1997). Consider a simple perceptron with two input units and one output unit that is an integration device (Ballard, 1986)—that is, it has a sigmoid-shaped activation function. Let one input unit represent the presence or absence of CS_1, and let the other input unit represent the absence or presence of CS_2. Furthermore, let the output unit represent whether a conditioned response (CR) has been elicited or not. This perceptron could be trained to make the desired responses in some discrimination learning task by applying a standard error-correcting learning rule such as the delta rule or the Widrow–Hoff rule. Standard machine learning rules such as these can be shown to be equivalent to fundamental mathematical accounts of animal learning. In particular, the Widrow–Hoff rule has been shown to be equivalent to the Rescorla–Wagner learning rule (Sutton & Barto, 1981).

On the other hand, perceptrons would also seem to be of little interest to animal learning researchers. This is because perceptrons are limited in the kinds of patterning classifications that they can make (Minsky & Papert, 1988). Traditional perceptrons cannot represent the stimulus–response pairings in some discrimination tasks that animals are able to learn. This point is elaborated below.

Consider again the perceptron with two input units and one output unit (Fig. 1A). Imagine a study in which we only wished this perceptron to generate a response when both input units were turned on. If both input units were off, or if only one input unit was on, then the output unit would not turn on. Fig. 1B (the AND problem) represents this desired behavior in a pattern space in which the coordinates of the input patterns are taken from the values of their corresponding input units. Note that a single line—more precisely, a single straight "cut" through the pattern space—can be used to separate the one pattern associated with a response (filled circle) from the three patterns that are not associated with a response (empty circles). Because a single line separates the off patterns from the on patterns in the space, this problem is described as being linearly separable. Perceptrons are capable

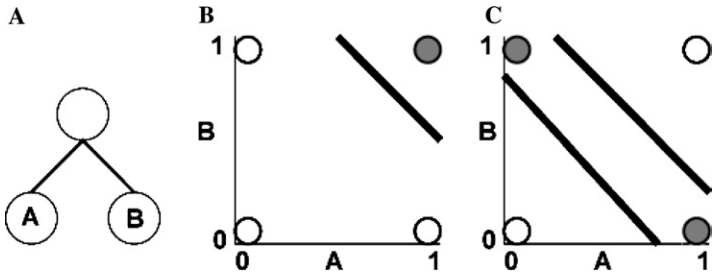

Fig. 1. (A) The structure of a simple perceptron with two input units (A and B) and one output unit. (B) The AND problem, which is linearly separable and can be learned by the simple perceptron. The perceptron learns to turn on to the one pattern (dark circle) and off to the other three patterns (outline circles). The diagonal line shows that one straight cut can separate the "on" pattern from the "off" patterns. (C) The XOR problem is not linearly separable, because two "cuts" (diagonal lines) are required to separate the "on" patterns from the "off" patterns. This problem cannot be learned by the perceptron represented in (A).

of representing solutions to any *linearly separable problem* (Minsky & Papert, 1988; Rosenblatt, 1962).

Imagine a second study in which we were interested in training a different set of responses. In this case, the output unit would only turn on if a single input unit were activated. If both input units were on at the same time, or if both were off at the same time, then the output unit would not respond. The pattern space for this problem (the exclusive-or or XOR problem) is provided in Fig. 1C. Note that in this figure two "cuts" are required to separate the on patterns from the off patterns. This is an example of a *linearly nonseparable problem*. Perceptrons are incapable of representing solutions to linearly nonseparable problems. However, animals and humans are capable of learning to solve such problems (e.g., Bellingham, Gillettebellingham, & Kehoe, 1985; Lachnit & Kimmel, 1993; Woodbury, 1943). In fact, such a problem can be solved by an invertebrate species (Deisig, Lachnit, Giurfa, & Hellstern, 2001; Schubert, Lachnit, Francucci, & Giurfa, 2002). As a result, an architecture as simple as the perceptron would seem to be of little interest to modern learning researchers.

Indeed, the inability of perceptrons to solve linearly nonseparable problems has been used as a strong argument that perceptrons do not provide valid theories of how animals learn to respond to combinations of stimuli. One learning paradigm that focuses upon stimulus combinations is the patterning experiment. In a patterning experiment, an animal learns to respond in one fashion to a single stimulus, and to respond in the opposite fashion when stimuli are combined. In positive patterning, the animal is trained not to respond to single stimuli and to respond to their conjunction [CS_{1-}, CS_{2-}, CS_1CS_{2+}]. In negative patterning, the animal is trained to respond to individual stimuli, and not to respond to a stimulus conjunction [CS_{1+}, CS_{2+}, CS_1CS_{2-}].

The perceptron that is illustrated in Fig. 1A represents one possible theory of patterning. Modern learning theorists such as Pearce (1997) have noted that a theory that can be expressed in this way is not powerful enough to account for negative patterning. The reason for making this claim is that negative patterning is equated with

the XOR problem, which we have already seen cannot be solved by a traditional perceptron, because it is a linearly nonseparable problem. "This is not a problem that is unique to this particular theory. There have been other attempts to develop single layer learning networks, and it has long been appreciated that they are unable to solve negative patterning discriminations, or, as it is more generally known, the *exclusive-or* problem" (Pearce, 1997, p. 131).

Going beyond the perceptron

Artificial neural networks have been argued to be of considerable relevance to the study of animal learning (Shanks, 1995). Yet some artificial neural networks, such as the perceptron, would appear to be inappropriate for such use. Instead, researchers typically use more powerful networks, such as the multilayer perceptron, to study animal learning in general, and to model phenomena like negative patterning in particular (e.g., Kehoe, 1988; Schmajuk & Dicarlo, 1992). One example of this is provided in a recent study of discrimination learning (Delamater, Sosa, & Katz, 1999).

Delamater et al. (1999) were interested in exploring the properties of a configural model of patterning in which configural representations emerged because of learning. As a result, they explored patterning using a multilayer artificial neural network that is in principle far more powerful than a perceptron (Rumelhart, Hinton, & Williams, 1986; for general introductions to such networks, see Dawson, under review; De Wilde, 1997; Ripley, 1996; Rojas, 1996). Their particular network had six different input units. Four of these were used to encode the presence of four different stimuli (A, B, C, or D). The other two were used to represent stimulus type. Both stimuli A and B were of type X. So, whenever either of these two stimuli was presented to the network, the input unit representing type X was also turned on. Similarly, stimuli C and D were of type Y; this was represented by also activating the sixth input unit whenever C or D was presented to the network. The network also had one output unit and four intermediate or 'hidden' units; all of these units employed the logistic activation function, which is a continuous approximation of the step function.

The hidden units provide this type of network more computational power than is available in the perceptron. Hidden units provide an opportunity to preprocess the activity of input units before signals are passed on to the output units. Hidden units can be thought of as having the ability to detect the presence of complex or higher-order features in the input patterns during this preprocessing. They can also be thought of as providing nonlinear transformations that are capable of "folding" a pattern space. In either case, the result is that the hidden units can transform a linearly nonseparable problem into one that is linearly separable, and which can therefore lead to appropriate responses from the output units (Rumelhart et al., 1986).

Delamater et al. (1999) used a multilayer perceptron because they wanted to explore the effect on patterning of representations that emerged in the intermediate layers of processors during a pre-training period. In the first phase of their experiment, their network was trained, using four different input patterns, to make discriminations between the four different individual stimuli (AX+, BX−, CY+, and DY−). In other words, it was reinforced (i.e., trained to activate) to stimuli A and C, and not

reinforced (i.e., trained to turn off) to stimuli B and D (see Table 1). With this pattern of responding, the network was discriminating, because it was generating different responses to the two X-type stimuli, as well as to the two Y-type stimuli. Once a network had learned to make these discriminations, it was placed in one of four different post-training conditions, each of which involved training the network to respond to three different input patterns.

Two of these conditions required the network to undergo a period of positive patterning, and are also illustrated in Table 1. In one, this positive patterning involved the stimuli that had been previously reinforced in the pre-training (AX−, CY−, and AXCY+). In the other, positive patterning was based on the stimuli that had not been previously reinforced (BX−, DY−, and BXDY+). Delamater et al. (1999) found that learning in the first condition was much faster than learning in the second condition, which indicated that previous reinforcement created internal representations that aided later positive patterning. The other two post-training conditions in their study involved negative patterning. In one, the negative patterning was based on the previously reinforced stimuli (AX+, CY+, and AXCY−). In the other, it was based on the stimuli that had not been previously reinforced (BX+, DY+, and BXDY−). Delamater et al. (1999) found that learning in the first condition was slower than learning in the second, which demonstrated that previous reinforcement created internal representations that hindered later negative patterning.

Table 1
The definition of the five different training sets that were used in the current simulations

Condition	A	B	C	D	X	Y	Output	Label
Pre-training	1	0	0	0	1	0	1	AX+
	0	1	0	0	1	0	0	BX−
	0	0	1	0	0	1	1	CY+
	0	0	0	1	0	1	0	DY−
	0	0	0	0	0	0	0	NULL
Positive patterning reinforced	1	0	0	0	1	0	0	AX−
	0	0	1	0	0	1	0	CY−
	1	0	1	0	1	1	1	AXCY+
	0	0	0	0	0	0	0	NULL
Positive patterning not reinforced	0	1	0	0	1	0	0	BX−
	0	0	0	1	0	1	0	DY−
	0	1	0	1	1	1	1	BXDY+
	0	0	0	0	0	0	0	NULL
Negative patterning reinforced	1	0	0	0	1	0	1	AX+
	0	0	1	0	0	1	1	CY+
	1	0	1	0	1	1	0	AXCY−
	0	0	0	0	0	0	0	NULL
Negative patterning not reinforced	0	1	0	0	1	0	1	BX+
	0	0	0	1	0	1	1	DY+
	0	1	0	1	0	1	0	BXDY−
	0	0	0	0	0	0	0	NULL

They are identical to those used by Delamater et al. (1999), with the exception that Delamater et al. did not include any of the patterns labeled "NULL."

What was particularly interesting about the Delamater et al. (1999) study was that after examining the performance of their networks, they proceeded to conduct a parallel animal learning study to determine whether pre-training affected animal patterning in the same way that it affected network patterning. In a pre-training phase, Sprague–Dawley rats learned to discriminate between two different sounds (tone vs. white noise) and between two different visual stimuli (steady light vs. flashing light). The rats then underwent a post-training patterning phase, in which they were placed in either a negative or a positive patterning paradigm, which involved either the stimuli that had been reinforced in the pre-training phase or the stimuli that had not been reinforced.

The results of the Delamater et al. (1999) animal study were quite different from the predictions made on the basis of the performance of their PDP network. First, for the rats there was strong evidence that previous reinforcement of stimuli aided negative patterning—a result that was completely opposite to the prediction made by the network. Second, there was at best weak evidence that previous reinforcement aided positive patterning. "The present data suggest that if changes in the internal representations of stimuli occur throughout training, they do not do so in the manner anticipated by the standard multi-layered network" (p. 108).

Is patterning really nonseparable?

Why would the multilayer network develop internal representations in the patterning experiment that were inconsistent with those that might have been developing in Delamater et al.'s (1999) animals? One possibility is that their network was far too powerful, and was in fact overfitting the data. A second possibility is that their network was not learning to make the same discriminations that were being made by the animals. Both of these possibilities are related to a crucial element of the study: when animal learning researchers typically define patterning, they do so in such a manner that it is actually linearly separable. As a result, a multilayer network is not required to learn the regularities of the patterning paradigm—a simple perceptron will do. The following paragraphs elaborate upon this issue.

Consider negative patterning. As was noted earlier, the standard representation of a negative patterning paradigm is [CS_{1+}, CS_{2+}, CS_1CS_{2-}]. Importantly, in this formulation, there are only three stimulus–response pairings. As a result, when these three conditions are plotted in a pattern space, the problem is linearly separable (see Fig. 2A), which is not consistent with the typical assumption that negative patterning is logically equivalent to XOR (e.g., Pearce, 1997). This is because a single cut separates the two "on" conditions from the single "off" condition.

Of course, it could be argued that the two representations described above both assume that there exists a fourth pattern for which no response is generated when both conditioned stimuli are absent. When this fourth pattern is added to the pattern spaces, positive patterning would remain linearly separable, but negative patterning would become linearly nonseparable. Clearly, when some learning researchers equate patterning with XOR (e.g., Pearce, 1997), they intend this interpretation to be true. However, this intention does not appear to be carried over into

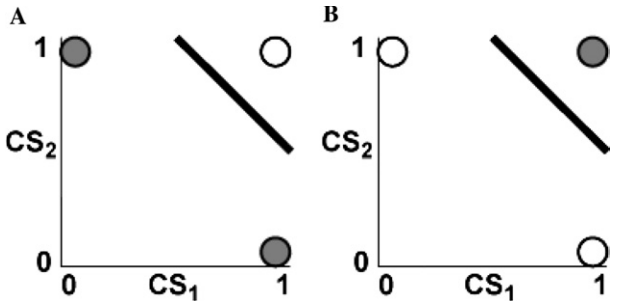

Fig. 2. (A) The pattern space for negative patterning as typically described by learning researchers. Note that it is linearly separable. (B) The pattern space for positive patterning.

simulation studies of patterning. For example, in Delamater et al.'s (1999) simulations each of the four post-training conditions had only three training patterns, and therefore each was logically equivalent to the linearly separable problems illustrated in Fig. 2. Because of this, a multilayer network was not required in their study. Because the problems that were presented to this network were linearly separable (see Fig. 2), it can easily be demonstrated that a simple perceptron can replace Delamater et al.'s entire multilayer network, and exhibits similar behavior to the larger network in the various conditions of the Delamater et al. experiment (Dawson, 2005, Chap. 16).

To correctly simulate the presence of experimental context, one must add an explicit training pattern that represents an intertrial interval in which all of the stimuli were absent, but experimental context was present. This can be accomplished by using a null pattern (see Table 1), in which none of the input units were turned on (representing the absence of experimental stimuli), and by training the network not to respond to this null pattern. Importantly, the addition of the null pattern changes the logical structure of the negative patterning training set. In particular, the addition of the null pattern would change the training sets in a fashion that is analogous to converting the appearance of Fig. 2A into the appearance of Fig. 1C, and make negative patterning logically equivalent to XOR.

Simulation 1: A multilayer perceptron and the null pattern

The network used in the original study by Delamater et al. (1999) was too powerful for the linearly nonseparable version of the patterning experiment that Delamater et al. originally conducted. However, it is appropriate for the version of the patterning experiment that includes the null pattern. This is because the addition of the null pattern makes the negative patterning conditions linearly nonseparable, and they are therefore beyond the capabilities of a mere perceptron. The purpose of the first simulation was to examine the performance of the multilayer network used by Delamater et al. on a new training set—the elaborated training sets that incorporated the null patterns, as detailed in Table 1.

Method

Network architecture and general training

The multilayer perceptrons that were used in Simulation 1 used six input units to represent the presence or absence of the six different conditioned stimuli, and used a single output unit to represent whether a conditioned response was generated or not. The network also included four hidden units. The output unit and the four hidden units were all integration devices that used the logistic activation function. All of the input units were connected to all of the hidden units, and all of the hidden units were connected to the output unit. There were no direction connections between input units and the output unit. It was trained with the generalized delta rule of Rumelhart et al. (1986).

Training sets

The training sets used, detailed in Table 1, were identical to those used by Delamater et al. (1999) with the exception that a single pattern, the null pattern, was added to each (i.e., to the pre-training and to each of the four post-training pattern sets).

General training procedure and experimental design

The networks in this first simulation study were trained with the generalized delta rule using a Visual Basic program called Rumelhart. This program is freely available from the following URL: http://www.bcp.psych.ualberta.ca/~mike/Book2/Software/Rumelhart/. The networks were trained with a learning rate of 0.5, with all of the weights and all of the unit biases (i.e., the "threshold" values of the output unit and each hidden unit) randomly selected from the range of −1 to +1. When a network was being trained on a training set, during one epoch of training it was presented each pattern in the training set once. The order of presentation was randomized each epoch. Connection weights and biases were updated after each pattern presentation. A momentum term was not used in the version of the generalized delta rule that was used to train these networks.

The experimental design for this simulation study was as follows: Each "subject" in the experiment was a multilayer perceptron. Prior to training, its initial structure was randomized. Then it was trained on the pre-training set of patterns until it converged. Without changing the weights of the network after this pre-training phase, it was then trained on one of the four post-training sets of patterns. The dependent measure was the number of epochs of training that the network required to converge during the post-training phase of the simulation. The four different post-training sets define a 2×2 factorial experiment, with the first factor being patterning condition (positive vs. negative) and the second factor being previous reinforcement (yes vs. no). Simulations were carried out until there were 10 different "subjects" in each cell of this experimental design. The question of interest was whether the behavior of the multilayer perceptrons in this study was different from the multilayer perceptrons originally studied by Delamater et al. (1999) on the linearly separable version of this simulation experiment.

Results and discussion

The top portion of Table 2 presents the average number of epochs of post-training required by networks to converge in each of the four conditions of the first simulation study. Analysis of variance of the data used to calculate these averages revealed that there was a main effect of type of patterning ($F_{1,36} = 183.22$, $p < .001$), and a significant interaction between type of patterning and type of reinforcement ($F_{1,36} = 8.48$, $p < .006$). However, there was no significant main effect of previous reinforcement ($F_{1,36} = 1.27$). As can be seen from examining Table 2, the results of this simulation were different once again from the results of the animal studies.

The main effect of patterning occurs because, regardless of the reinforcement condition, positive patterning was learned substantially faster than was negative patterning. The significant interaction between type of patterning and type of reinforcement, combined with no significant effect of previous reinforcement, occurs because previous reinforcement has a different effect on positive patterning than it does on negative patterning. For positive patterning, the previous reinforcement of stimuli leads to a moderate, but statistically significant, increase in the speed of learning ($t_9 = 5.42$, $p < .006$). In contrast, for negative patterning the previous reinforcement of stimuli leads to a moderate, but statistically significant, decrease in the speed of learning ($t_9 = 7.13$, $p < .006$).

This pattern of results is very similar to the pattern that Delamater et al. (1999) observed in their multilayer network. On the one hand, this is encouraging, because it indicates that the results obtained from their network would not have been dramatically altered by including the null pattern in each of the training sets used in their simulation. On the other hand, this is disappointing, because it also means that the network trained in our first simulation did not generate the pattern of results that Delamater et al. observed in their animal experiments. Recall that in their experiments, previous reinforcement improved learning in the negative patterning paradigm. In the next section, a second simulation is explored to determine whether this unfortunate pattern of results is inevitable with multilayer networks. In this simulation, the training sets still include the null pattern, and the network is identical to the one used by Delamater et al. with one exception: the output unit is changed from an integration

Table 2
Average number of epochs (with standard deviations) for pre-trained multilayer perceptrons to converge to solutions to patterning problems in the two simulations

	Positive patterning	Negative patterning
Networks with an output integration device		
Previously reinforced	538.30 (50.99)	2572.50 (528.67)
Not previously reinforced	759.00 (176.81)	2072.90 (546.31)
Networks with an output value unit		
Previously reinforced	799.10 (128.73)	1082.40 (332.13)
Not previously reinforced	1031.90 (110.83)	3166.20 (1022.04)

Each cell represents an average of 10 different simulations. The top part of the table presents the results for the networks in the first simulation, while the bottom part of the table provides the results for the networks in the second simulation.

device into a different kind of processor, one that activates to only a narrow range of input signals. Ballard (1986) has called this kind of processor a value unit.

Simulation 2: A hybrid multilayer perceptron and the null pattern

Delamater et al. (1999) hypothesized that one reason that may account for the observed learning dynamics of their animals is that pre-training does not merely affect the internal representations that are constructed by the learning systems (as it does in their simulations). Instead, pre-training might also affect animal learning by altering the amount of processing that stimuli receive. "For example, suppose that reinforced stimuli are processed more effectively than nonreinforced stimuli. One consequence of this would be to enhance the salience of the stimuli, and this could, in turn, result in faster learning" (p. 109).

Delamater et al. (1999) point out that this possibility is consistent with the attentional theory of learning formalized by Mackintosh (1975). In the Rescorla–Wagner model, the change in associative strength (ΔV_A) of some CS_A is defined as $\Delta V_A = k(\lambda - \Sigma V)$, where k is the learning rate, λ is the maximum amount of associative strength that can be supported by the UCS, and ΣV is the total amount of associative strength for all stimuli that are present. Mackintosh adapted this rule in two ways. First, he argued that the attention devoted to a stimulus during learning should vary depending upon how well the stimulus predicts reinforcement: the organism learns to pay more attention to a good predictor, and to pay less attention to a poor predictor. Second, he argued that changes in associative strength depend not on the summed effects of all stimuli, but rather on the predictive power of each unique cue.

Mackintosh's (1975) learning model can be expressed as $\Delta V_A = \alpha_A k(\lambda - V_A)$, where V_A is the associative strength of CS_A, α_A is a constant that indicates the amount of attention that is devoted to CS_A, and ΔV_A, k, and λ have the same interpretation as in the Rescorla–Wagner rule above. Mackintosh also provided rules that indicated how the value of α_A would change as a function of learning the predictive power of CS_A. Krushke (e.g., 2001, 2003) has shown how the Mackintosh rule can be subsumed in a connectionist network in which modifiable attentional gates work in concert with connection weights to control the degree to which different stimuli affect the output units in the network.

Krushke's (e.g., 2001, 2003) is not the only approach that one could use to differentiate the processing of different stimuli. In Krushke's networks, and in Mackintosh's (1975) learning model, there are quantitative differences in the way that stimuli are processed: CS_A could be processed more effectively than CS_B by setting the constant α_A to a higher value than the constant α_B. An alternative approach would be to enforce qualitative differences between the processing of different stimuli. For example, in some connectionist networks when a network is presented stimuli that are reinforced, a learning rule is used to strengthen weights. However, when nonreinforced stimuli are presented, the learning rule is not invoked, but instead the network's weights decay (e.g., Shepard & Kannappan, 1991; Weisman et al., 1998).

This latter approach to differentiating the processing of stimuli is built directly into Dawson and Schopflocher's (1992) learning rule for a different kind of processor, called value units. Value units use an activation function that generates maximum activity when the net input is equal to the mean (μ) of a Gaussian equation that has a standard deviation of 1. This means that activity quickly drops to zero when the net input becomes somewhat larger or smaller than μ. The paragraphs below explain why value units process stimuli differentially when they are trained.

In the generalized delta rule for multilayer perceptrons (Rumelhart et al., 1986), connection weights are changed to minimize the following definition of network error $E = 1/2\ \Sigma\Sigma(t_{pj} - a_{pj})^2$, where t_{pj} is the target or desired value of output unit j when some input pattern p is presented to the network, a_{pj} is the observed value of output unit j when pattern p is presented, and error is summed over all of the output units in the network and over all of the patterns that are being presented to it.

In contrast, the learning rule for value units defines network error as $E = 1/2\ \Sigma\Sigma(t_{pj} - a_{pj})^2 + 1/2\ \Sigma\Sigma\ t_{pj}\ (\mathrm{net}_{pj} - \mu_j)^2$. The first part of this equation is identical to network error as defined for the generalized delta rule. The second part of the equation is an additional heuristic cost function that Dawson and Schopflocher used to speed up learning, and to prevent value unit networks from being stranded in a particular type of local minimum during learning. It assumes that the value of t_{pj} will be either 1 or 0. When t_{pj} has a value of 1, the additional error term indicates that the net input to unit j that is produced by pattern p (net_{pj}) should be equal to the mean of the Gaussian activation function (μ_j). If this is not the case, then this is an additional source of error that can be used to guide the modification of connection weights. When t_{pj} has a value of 0, the additional error term falls out of the calculation, and the learning rule essentially reverts to the standard generalized delta rule.

The importance of Dawson and Schopflocher's (1992) definition of network error is that it results in differential processing of stimuli. That is, the definition of error for reinforced stimuli ($t_{pj} = 1$) includes more information than does the definition of error for stimuli that are not reinforced ($t_{pj} = 0$). In general terms, this definition of error takes advantage of additional knowledge about reinforced stimuli (i.e., the fact that their net input should be near the mean of the Gaussian) to speed up learning; this additional information is not available for nonreinforced input patterns. In specific relation to the current paper, this also suggests that the Dawson and Schopflocher learning rule can be used as one procedure to explore Delamater et al.'s (1999) hypothesis that the results of their animal experiments are due to differential processing of reinforced and nonreinforced stimuli. That is, if we take the network that was used in the previous simulation, and replace its output unit with a value unit (but leave all other units the same as in the previous simulation), then one consequence is that differential processing of stimuli will be implemented automatically. The question of interest is whether this change in network architecture will lead to a change in performance when compared to the previous network.

Method

Network architecture and general training

The multilayer perceptrons that were used in this second simulation study were identical to those used in the first simulation study, with the exception that the output unit was changed from an integration device into a value unit. Because of this change, they networks were trained with a learning rule for multilayer value unit networks developed by Dawson and Schopflocher (1992). This learning rule is an extension of the generalized delta rule that was originated by Rumelhart et al. (1986). Dawson and Schopflocher demonstrated that this rule can be used to train hybrid multilayer perceptrons (i.e., networks like the one in this study where the output unit is a value unit, and the hidden units are integration devices).

Training sets

The training sets were identical to those used in first simulation study, and are detailed in Table 1.

General training procedure and experimental design

The networks in this second study were trained with Dawson and Schopflocher's (1992) variation of Rumelhart et al. (1986) generalized delta rule. This learning rule is an option in the program that was used to train the networks in the first study. The networks were trained with a learning rate of 0.1, with all of the weights and all of the unit biases randomly selected from the range of −1 to +1. When a network was being trained on a training set, during one epoch of training it was presented each pattern in the training set once. The order of presentation was randomized each epoch. Connection weights and biases were updated after each pattern presentation. A momentum term was not used in the training of these networks. The experimental design of this second study was identical to the design of the first. The question of interest in this study was whether the behavior of the hybrid multilayer perceptrons in this second study was different from the behavior of the multilayer perceptrons in the first simulation because of the change in the activation function of the output unit.

Results and discussion

The bottom portion of Table 2 presents the average number of epochs of post-training required by networks to converge in each of the four conditions of the second simulation study. Analysis of variance of the data used to calculate these averages revealed that there was a significant main effect of type of patterning ($F_{1,36} = 49.38$, $p < .001$), a significant main effect of type of reinforcement ($F_{1,36} = 45.34$, $p < .001$), and a significant interaction between these two factors ($F_{1,36} = 28.94$, $p < .001$).

First, the main effect of patterning occurs because, regardless of the reinforcement condition, positive patterning was learned substantially faster than was negative patterning. This result replicates one of the main findings of the first simulation study, and is again more consistent with the results of animal experiments.

Second, the main effect of type of reinforcement represents the fact that, on average, conditions in which stimuli have been previously reinforced lead to faster learning than do conditions in which stimuli have not been previously reinforced. Importantly, this main effect must be interpreted in the context of the significant interaction that was found in the results of this experiment. This is because the effect of previous reinforcement is, as can be seen from Table 2, far more effective for negative patterning than it is for positive patterning. For negative patterning, previous reinforcement led to an average speed up of learning of 2083.8 epochs ($t_9 = 7.134$, $p < .001$). For positive patterning, previous reinforcement results in learning speeding up by an average of 232.8 epochs, which was statistically significant ($t_9 = 5.442$, $p < .001$) but considerably less of an improvement than was noted for negative patterning.

The pattern of results for this second experiment is particularly encouraging these results are the most similar to those reported by Delamater et al. (1999) for their animal experiments. In those experiments, previous reinforcement produced markedly improved performance for negative patterning, but had much less of an effect on positive patterning.

General discussion

In the current paper, it was hypothesized that the discrepancy that Delamater et al. (1999) observed between their animal results and their simulation results might be due to particular design decisions that they made in creating their connectionist networks. In particular, we noted that the training sets that Delamater et al. used in their simulations did not define a linearly nonseparable problem. To correct this, we added a null pattern to each of the five training sets that were originally described by Delamater et al. (1999).

In our first simulation study, we used the same architecture used by Delamater et al. (1999) in the patterning experiments: a multilayer perceptron with four hidden units and one output unit. All of these units were integration devices; that is, they all used the logistic activation function to convert their net input into internal activation. We found that negative patterning was much harder to learn than was positive patterning, which is a result that is consistent with animal learning. However, the addition of the null pattern to the training sets used to train this network did not change the other main results that were originally obtained by Delamater et al. Previous reinforcement aided positive patterning, and hindered negative patterning, which is not consistent with the animal results.

In our second simulation, we explored a possibility that Delamater et al. (1999) suggested: the effects of pre-training may not only develop internal representations, but might also alter the efficiency with which different stimuli are presented. In particular, it could be the case that the reinforcement of stimuli leads to more effective learning. We explored this possibility by altering the network that was used in Simulation 1. We still used four hidden units that were integration devices. However, we converted the output unit from an integration device into a value unit. The rea-

son for this was that the learning rule for an output value unit uses a more informative error term for reinforced stimuli than for nonreinforced stimuli, resulting in differential treatment of these types of patterns (Dawson & Schopflocher, 1992). In this second simulation, we found that negative patterning was still much harder to learn than was positive patterning. We also observed that while previous reinforcement facilitated learning under positive patterning, it had a much stronger facilitation to learning under negative patterning. Qualitatively, these results provide a better fit to Delamater et al.'s animal experiments. One of the key findings revealed in the two simulations was that for this particular study one could produce patterns of learning in networks that were more similar to those observed in animals by changing the output unit from an integration device into a value unit. One reason for this is that the learning rule for value units implicitly treats reinforced stimuli differently than nonreinforced stimuli. This raises another question: might the contributions provided by value units in the current paper extend to other learning phenomena?

While answering this question is an ongoing research program, other results that we have obtained do suggest that the value unit architecture is capable of making more general contributions to learning theory.

For example, one of the most surprising and theoretically important effects in associative learning is the overexpectation effect. The effect is produced when two conditioned stimuli (CSs), A and X, are independently paired with a given unconditioned stimulus (US) until asymptotic learning occurs, and then A and X are presented in compound and paired with the US. The result is that responding to A or X is reduced following AX–US pairings relative to a control condition in which no compound training is given. This result is intuitively surprising because A and X apparently lost associative strength despite continued reinforcement during the compound training. Nevertheless, overexpectation effects have been found in studies on Pavlovian fear conditioning in rats (e.g., Blaisdell, Denniston, & Miller, 2001; Kremer, 1978; Rescorla, 1970), appetitive conditioning in rats (Lattal & Nakajima, 1998; Rescorla, 1999), and autoshaping with pigeons (Khallad & Moore, 1996). Moreover, the effect can be reversed by naloxone injections, suggesting that it is modulated by the opioid system (McNally, Pigg, & Weidemann, 2004).

Although counter-intuitive, the overexpectation effect was perfectly predicted by the Rescorla–Wagner model of associative learning (Rescorla & Wagner, 1972). Presumably, this should in turn mean that the overexpectation effect should be found in perceptrons, because the error-correcting methods used to train perceptrons have been proven to be equivalent to the Rescorla–Wagner rule (Sutton & Barto, 1981). However, Dawson and Spetch (under revew) demonstrated that perceptrons are unable to produce the overexpectation effect when their output unit is an integration device. However, if the output unit is changed into a value unit, then the overexpectation effect is produced in the simulation. We are currently investigating other specific contributions of the value unit architecture, and are attempting to formulate a general account of learning in networks that attempts to relate properties of the activation function to traditional learning phenomena.

Acknowledgments

The research reported in the manuscript was supported by NSERC and SSHRC research grants awarded to M.R.W.D.

References

Ballard, D. (1986). Cortical structures and parallel processing: Structure and function. *The Behavioral And Brain Sciences, 9*, 67–120.

Bellingham, W. P., Gillettebellingham, K., & Kehoe, E. J. (1985). Summation and configuration in patterning schedules with the rat and rabbit. *Animal Learning & Behavior, 13*, 152–164.

Blaisdell, A. P., Denniston, J. C., & Miller, R. R. (2001). Recovery from the overexpectation effect: Contrasting performance-focused and acquisition-focused models of retrospective revaluation. *Animal Learning & Behavior, 29*, 367–380.

Dawson, M. R. W. (2004). *Minds And Machines: Connectionism And Psychological Modeling*. Malden, MA: Blackwell.

Dawson, M. R. W. (under review). *Connectionism: A hands-on approach*. Malden, MA: Blackwell.

Dawson, M. R. W., & Schopflocher, D. P. (1992). Modifying the generalized delta rule to train networks of nonmonotonic processors for pattern classification. *Connection Science, 4*, 19–31.

Dawson, M.R.W. & Spetch, M.L. (under review). Traditional perceptrons do not produce the overexpectation effect. *Psychonomic Bulletin*, Review.

De Wilde, P. (1997). *Neural network models* (2nd ed). London: Springer.

Deisig, N., Lachnit, H., Giurfa, M., & Hellstern, F. (2001). Configural olfactory learning in honeybees: Negative and positive patterning discrimination. *Learning & Memory, 8*, 70–78.

Delamater, A. R., Sosa, W., & Katz, M. (1999). Elemental and configural processes in patterning discrimination learning. *The Quarterly Journal Of Experimental Psychology B, 52*, 97–124.

Kehoe, E. J. (1988). A layered network model of associative learning: Learning to learn and configuration. *Psychological review, 95*, 411–433.

Khallad, Y., & Moore, J. (1996). Blocking, unblocking, and overexpectation in autoshaping with pigeons. *Journal of the Experimental Analysis of Behavior, 65*, 575–591.

Kremer, E. F. (1978). Rescorla–Wagner model—Losses in associative strength in compound conditioned stimuli. *Journal of Experimental Psychology-Animal Behavior Processes, 4*, 22–36.

Krushke, J. K. (2001). Toward a unified model of attention in associative learning. *Journal of Mathematical Psychology, 45*, 812–863.

Krushke, J. K. (2003). Attention in learning. *Current Directions In Psychological Science, 12*, 171–175.

Lachnit, H., & Kimmel, H. D. (1993). Positive and negative patterning in human classical skin-conductance response conditioning. *Animal Learning & Behavior, 21*, 314–326.

Lattal, K. M., & Nakajima, S. (1998). Overexpectation in appetitive Pavlovian and instrumental conditioning. *Animal Learning & Behavior, 26*, 351–360.

Mackintosh, N. J. (1975). A theory of attention: Variation in the associability of stimuli with reinforcement. *Psychological Review, 82*, 276–298.

McNally, G. P., Pigg, M., & Weidemann, G. (2004). Blocking, unblocking, and overexpectation of fear: A role for opioid receptors in the regulation of Pavlovian association formation. *Behavioral Neuroscience, 118*, 111–120.

Minsky, M., & Papert, S. (1988). *Perceptrons* (3rd ed). Cambridge, MA: MIT Press.

Pearce, J. M. (1997). *Animal learning and cognition: An introduction*. East Sussex: Psychology Press.

Rescorla, R. A. (1970). Reduction in effectiveness of reinforcement after prior excitatory conditioning. *Learning and Motivation, 1*, 372–381.

Rescorla, R. A. (1999). Summation and overexpectation with qualitatively different outcomes. *Animal Learning & Behavior, 27*, 50–62.

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A. H. Black & W. F. Prokasy (Eds.), *Classical conditioning II: Current research and theory* (pp. 64–99). New York, NY: Appleton-Century-Crofts.

Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge, UK: Cambridge University Press.

Rojas, R. (1996). *Neural networks: A systematic exploration*. Berlin: Springer.

Rosenblatt, F. (1962). *Principles of neurodynamics*. Washington: Spartan Books.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*, 533–536.

Schmajuk, N. A., & Dicarlo, J. J. (1992). Stimulus configuration, classical-conditioning, and hippocampal function. *Psychological Review, 99*, 268–305.

Schubert, M., Lachnit, H., Francucci, S., & Giurfa, M. (2002). Nonelemental visual learning in honeybees. *Animal Behaviour, 64*, 175–184.

Shanks, D. R. (1995). *The psychology of associative learning*. Cambridge, UK: Cambridge University Press.

Shepard, R. N., & Kannappan, S. (1991). Connectionist implementation of a theory of generalization. In R. P. Lippmann, J. Moody, & D. S. Touretzky (Eds.), *Advances in neural information processing Systems 3* (pp. 1–7). San Mateo, CA: Morgan Kauffman.

Sutton, R. S., & Barto, A. G. (1981). Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological Review, 88*, 135–170.

Weisman, R., Njegovan, M., Sturdy, C., Phillmore, L., Coyle, J., & Mewhort, D. (1998). Frequency-range discriminations: Special and general abilities in zebra finches (*Taeniopygia guttata*) and humans (*Homo sapiens*). *Journal of Comparative Psychology, 112*, 244–258.

Woodbury, C. B. (1943). The learning of stimulus patterns by dogs. *Journal of Comparative Psychology, 35*, 29–40.