Neural Computation, 14(9):2157–79, 2002.

Pre-integration lateral inhibition enhances unsupervised learning

M. W. Spratling and M. H. Johnson
Centre for Brain and Cognitive Development, Birkbeck College, London. UK.

Abstract

A large and influential class of neural network architectures use post-integration lateral inhibition as a mechanism for competition. We argue that these algorithms are computationally deficient in that they fail to generate, or learn, appropriate perceptual representations under certain circumstances. An alternative neural network architecture is presented in which nodes compete for the right to receive inputs rather than for the right to generate outputs. This form of competition, implemented through pre-integration lateral inhibition, does provide appropriate coding properties and can be used to efficiently learn such representations. Furthermore, this architecture is consistent with both neuro-anatomical and neuro-physiological data. We thus argue that pre-integration lateral inhibition has computational advantages over conventional neural network architectures while remaining equally biologically plausible.

1 Introduction

The nodes in a neural network generate activity in response to the input they receive. This response can be considered to be a representation of the input stimulus. The patterns of neural activity which constitute efficient and complete representations will vary between tasks. Therefore, for a neural network algorithm to be widely applicable it must be capable of generating a variety of response patterns. Such a neural network needs to be able to respond to individual stimuli as well as multiple stimuli presented simultaneously, to distinguish overlapping stimuli, and to deal with incomplete stimuli. Many highly influential and widely used neural network algorithms meet some, but not all, of these criteria. We present a simple neural network which does meet all of these criteria. It succeeds in doing so by using a novel form of lateral inhibition.

Lateral inhibition is an essential feature of many artificial, self-organizing, neural networks. It provides a mechanism through which neurons compete for the right to generate a response to the current pattern of input activity. This not only makes responses more selective (in the short-term), but since learning is commonly activity-dependent it also makes the receptive fields of individual nodes more distinct (in the long-term). Competition thus enables nodes to represent distinct patterns within the input space. Lateral inhibition provides competition by enabling nodes to inhibit each other from generating a response. This form of competition is used in numerous neural network algorithms. It is implemented either explicitly through lateral connections between nodes (Földiák, 1989, 1990; Marshall, 1995; Sirosh and Miikkulainen, 1994; Swindale, 1996; von der Malsburg, 1973; Oja, 1989; Sanger, 1989; O'Reilly, 1998) or implicitly through a selection process which chooses the 'winning' node(s) (Rumelhart and Zipser, 1985; Kohonen, 1997; Grossberg, 1987; Ritter et al., 1992; Hertz et al., 1991; Földiák, 1991; Wallis, 1996). Despite the large diversity of implementations, these algorithms share a common mechanism of competition. A node's success in this competition is dependent on the total strength of the stimulation it receives and nodes which compete unsuccessfully have their output activity suppressed. This class of architectures can thus be described as implementing 'post-integration lateral inhibition'.

Single-layer neural networks which make use of this competitive mechanism fail to meet all of the criteria set out above. These representational deficiencies can be overcome by using a neural network architecture in which there is inhibition of inputs rather than of outputs (figure 1(b)). In our algorithm each node attempts to 'block' its preferred inputs from activating other nodes. Nodes thus compete for the right to receive inputs rather than for the right to generate outputs. We describe this form of competition as 'pre-integration lateral inhibition'. Such a network can be trained using simple, activity-dependent, learning rules. Since learning is a function of activation, improved response properties result in more correct learning episodes (Marshall, 1995), and hence, the advantageous coding properties that arise from pre-integration lateral inhibition result in efficient, unsupervised, learning. We demonstrate the abilities of a network using pre-integration lateral inhibition with the aid of two simple tasks. These tasks provide an easily understood illustration of the properties required in order to learn useful representations, and have been used previously to demonstrate the pattern recognition abilities required by models of the human perceptual system (Marshall, 1995; Marshall and Gupta, 1998; Nigrin, 1993; Földiák, 1990; Hinton et al., 1995).

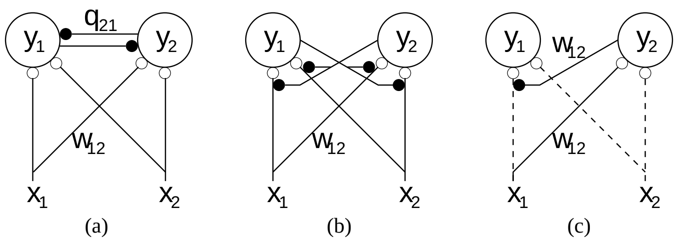


Figure 1: Models of lateral inhibition. Nodes are shown as large circles, excitatory synapses as small open circles and inhibitory synapses as small filled circles. (a) The standard model of lateral inhibition provides competition between outputs. (b) The pre-integration lateral inhibition model provides competition for inputs. (c) The pre-integration lateral inhibition model with only one lateral weight shown. This lateral weight has an identical value (w_{12}) to the afferent weight shown as a solid line.

2 Method

Simple, two-node, neural networks competing via post-integration lateral inhibition and pre-integration lateral inhibition are shown in figure 1. Pre-integration lateral inhibition enables each node to inhibit other nodes from responding to the same inputs. Hence, if a node is active and it has a strong synaptic weight to a certain input then it should inhibit other nodes from responding to that input. A simple implementation of this idea, for a two-node network, would be:

$$y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i - \alpha w_{i2}y_2\right)^+$$

$$y_2 = \sum_{i=1}^{m} \left(w_{i2}x_i - \alpha w_{i1}y_1\right)^+ .$$

Where y_j is the activation of node j, w_{ij} is the synaptic weight from input i to node j, x_i is the activation of input i, α is a scale factor controlling the strength of lateral inhibition, and $(v)^+$ is the positive half-rectified value of v. These simultaneous equations can be solved iteratively. To help ensure that a steady-state solution is reached it has been found useful to gradually increase the value of α at each iteration from an initial value of zero. Physiologically, this regime might correspond to a delay in the build up of lateral inhibition due to additional propagation delays via inhibitory interneurons. An equivalent implementation of post-integration lateral inhibition would be:

$$y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i\right) - \alpha q_{21}y_2$$

$$y_2 = \sum_{i=1}^{m} \left(w_{i2}x_i\right) - \alpha q_{12}y_1 .$$

Where q_{kj} is the lateral weight from node k to node j. Note that for pre-integration lateral inhibition rectification is applied to the inhibited inputs prior to summation. This is essential otherwise pre-integration lateral inhibition would be mathematically equivalent to post-integration lateral inhibition. For example, without rectification the equation given above for pre-integration lateral inhibition to node 1 would become $y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i - \alpha w_{i2}y_2\right)$ which can be rewritten as $y_1 = \sum_{i=1}^{m} \left(w_{i1}x_i\right) - \alpha y_2 \sum_{i=1}^{m} w_{i2}$ which is equivalent to the equation for post-integration lateral inhibition when $\sum_{i=1}^{m} w_{i2} = q_{21}$. In the discussion section we suggest that pre-integration lateral inhibition is achieved in the cortex via inhibitory contacts on the dendrites of excitatory cells. Due to the nonlinear response properties of dendrites (Mel, 1994; Koch et al., 1983; Shepherd and Brayton, 1987) such dendritic inhibition will have effects confined to the local region of the dendritic tree and little or no impact on excitatory inputs to other branches of the dendrite. This nonlinearity is achieved in our model by using rectification.

For pre-integration lateral inhibition the magnitudes of the lateral weights are identical to the corresponding afferent weights. For example, the activation of node 2 inhibits input i of node 1 with a strength weighted by w_{i2}. This lateral weight is identical to the afferent weight node 2 receives for input i (see figure 1(c)). The values of the afferent weights provide appropriate lateral weights since the objective of pre-integration lateral inhibition is to allow an active node with a strong synaptic weight to a certain input to strongly inhibit other nodes from responding to that same input, while not inhibiting those inputs from which it receives weak weights. Since the lateral weights have identical values to afferent weights it is not necessary to determine their strengths separately.

In a more biologically plausible version of the model separate lateral weights could be learnt independently. This would be possible since weights that have identical values also have identical (but reversed) pre- and post-synaptic activation values. For example, in figure 1(c) the afferent weight w_{12} connects input 1 to node 2 and an identical lateral weight connects node 2 to input 1 (at node 1). Hence, by using similar activity-dependent learning rules, and identical initial values, appropriate lateral weights could be learnt.

While the equations presented above provide a simple illustration of pre-integration lateral inhibition a more complex formulation is required for application to neural networks containing arbitrary numbers of nodes (n) and receiving arbitrary numbers of inputs (m). In a network containing more than two nodes each input could be inhibited by multiple lateral connections. What strength of inhibition should result from multiple nodes? One possibility would be to sum the effects from all the inhibitory connections. However, simply changing the number of nodes in such a network could have a significant effect on the total inhibition. To ensure that inhibition is not a function of n we inhibit each node by the maximum value of the inhibition generated by all the nodes. Hence, the activation of each node is calculated as:

$$y_j = \sum_{i=1}^{m} \left(w_{ij} x_i - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \{ w_{ik} y_k \} \right)^{+} .$$

Note that nodes do not inhibit their own inputs. Early in training nodes tend to have small, undifferentiated, weights and hence small activation values. This results in weak lateral inhibition that permits all nodes to respond to each input pattern. To ensure that lateral inhibition is equally effective at the start of training, as at the end, the strength of inhibition is normalized as follows: the lateral weight is divided by the maximum lateral weight originating from the inhibiting node and the inhibiting node's activity is divided by the maximum activity of all the nodes in the network:

$$y_j = \sum_{i=1}^{m} \left(w_{ij} x_i - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ \frac{w_{ik}}{\max_{l=1}^{m} \{ w_{lk} \}} \frac{y_k}{\max_{l=1}^{n} \{ y_l \}} \right\} \right)^{+} .$$

Normalizing the strength of lateral inhibition ensures that there is strong competition resulting in a differentiation in activity values and more rapid activity-dependent learning. Normalization also results in the strength of inhibition being constrained to be in the range zero to one. However, this form of inhibition would have a greater affect on a stimulus presented at low contrast (using smaller values of x_i) than on the same stimulus presented at high contrast. Tolerance to changes in stimulus contrast is provided by modifying the equation as follows:

$$y_j = \sum_{i=1}^{m} w_{ij} x_i \left(1 - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ \frac{w_{ik}}{\max_{l=1}^{m} \{ w_{lk} \}} \frac{y_k}{\max_{l=1}^{n} \{ y_l \}} \right\} \right)^{+} . \quad (1)$$

This formulation was used of produce all the results presented in this paper. The value of α was increased from zero to four in steps of 0.25. Activation values generally reached a steady-state at lower alpha, in which case competition was terminated before α reached its maximum value. The change in the value of α between iterations was found to be immaterial to the final activation values provided it was less than 0.5.

Networks were trained using unsupervised learning. Weights were initially set all equal to $\frac{1}{m}$ (we describe such nodes as 'uncommitted'). (The algorithm works equally well with randomly initialized weights.) In order to cause differentiation of the receptive fields noise was added to the node activations at each iteration during the competition. Noise values were random numbers uniformly distributed in the range 0 to 0.001. When several nodes with identical weights are equally activated by the current input, noise should cause a single node from this identical group to win the competition and hence respond to that input. This bifurcation of activity values requires one node to have a significantly higher activation (after adding noise), than all other nodes in the identical group, so that it can successfully inhibit the other nodes. As the number of nodes increases it becomes less likely that a single node will receive a significantly higher noise value than the others. Hence, this noisy selection process was found to be more robust when noise was added to a subset of nodes: noise was added to each node with probability $\frac{4}{n}$. (Zero-mean, Gaussian-distributed noise, added to the activity values of all nodes, has also been used successfully.) Since the magnitude of the noise is small it has no effect on competition once nodes have begun to learn preferences for distinct input patterns.

Synaptic weights were modified at completion of the competition phase (*i.e.*, using the final values for y_j found after iteration of equation 1). The following learning rule was employed:

$$\Delta w_{ij} = \beta \; \frac{(x_i - \bar{x})}{\sum_{k=1}^{m} x_k} \; \frac{(y_j - \bar{y})^{+}}{\sum_{k=1}^{n} y_k} . \quad (2)$$

Where $\bar{x}$ is the mean of the input activations (*i.e.*, $\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$), and $\bar{y}$ is the mean of the output activations (*i.e.*, $\bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_j$), and β is a parameter controlling the learning rate. Following learning, synaptic weights were clipped at zero such that $w_{ij} = (w_{ij})^+$ and were normalized such that $\sum_{i=1}^{m} (w_{ij})^+ = 1$.

This learning rule encourages each node to learn weights selective for a set of coactive inputs. This is achieved since when a node is more active than average it increases its synaptic weights to active inputs and decreases its weights to inactive inputs. Hence, only sets of inputs which are consistently coactive will generate strong afferent weights. In addition, the learning rule is designed to ensure that nodes can represent stimuli which share input features in common (*i.e.*, to allow the network to represent overlapping patterns). This is achieved by rectifying the post-synaptic term of the rule so that no weight changes occur when the node is less active than average. If learning was not restricted in this way whenever a pattern was presented all nodes which represented patterns with overlapping features would reduce their weights to these features. Our learning rule is similar to the instar rule (Grossberg, 1976, 1978), except that only nodes that are more active than average have their weights modified. Hence, post-synaptic activation exceeding a threshold is required before synaptic modification is enabled. The learning rule is also similar to the covariance rule (Sejnowski, 1977; Grzywacz and Burgi, 1998), except the post-synaptic term is only allowed to be greater than or equal to zero. Hence, sub-threshold post-synaptic activity does not result in any changes in synaptic strength. Normalization by the total of the input and output activities helps to ensure that each stimulus contributes equally to learning and that the total weight change (across all nodes) at each iteration is similar. Division by zero errors were avoided by learning only when the maximum input activity exceeded an arbitrary threshold (0.1).

Nodes learn to respond to specific features within the input space. In order to represent stimuli which contain multiple features the activation function allows multiple nodes to be active simultaneously. With overlapping patterns this could allow the same stimulus to be represented in multiple ways. For example, consider a simple network receiving input from three sources (labelled 'a', 'b' and 'c') which has the task of representing three patterns: 'a', 'ab', and 'bc'. Individual nodes in a network may learn to become selective to each of these patterns. Each individual pattern would thus be represented by the activity of a single node. However, it would also be possible for a network in which nodes had learnt patterns 'a' and 'bc' to respond to pattern 'ab' by fully activating the node selective to 'a' and partially activating the node selective to 'bc'. Pattern 'ab' could be the result of coactivation of pattern 'a' and a partially occluded version of pattern 'bc'. However, if 'ab' occurs with similar frequency to the other two patterns then it would seem more likely that 'ab' is an independent input pattern that should be represented in a similar way to the other patterns (*i.e.*, by the activation of a specific node tuned to that pattern). To ensure that in such situations the network learns all input patterns it is necessary to introduce a second learning rule:

$$\Delta w_{ij}^{-} = -\beta^{-} \, (x_i - X_{ij}) \, (y_j - \bar{y}) \,. \tag{3}$$

Where β^{-} is a parameter controlling the learning rate, and X_{ij} is the input activation from source i to node j after inhibition, *i.e.*,

$$X_{ij} = x_i \left(1 - \alpha \max_{\substack{k=1 \\ (k \neq j)}}^{n} \left\{ \frac{w_{ik}}{\max_{l=1}^{m} \{w_{lk}\}} \frac{y_k}{\max_{l=1}^{n} \{y_l\}} \right\} \right)^{+} .$$

The values of X_{ij} are determined from equation 1. Negative weights were constrained such that $0 \geq \sum_{i=1}^{m} (w_{ij})^{-} \geq -1$. This learning rule is only applied to synapses which have a weight of zero (or less than zero) caused by application of the learning rule given in equation 2 (or prior application of this rule). Negative weights are generated when a node is active and inputs, which are not part of the nodes' preferred pattern, are inhibited. This can only occur when multiple nodes are coactive. If the pattern, to which this set of coactive nodes are responding, re-occurs then the negative weights will grow. When the negative weights are sufficiently large the response of these nodes to this particular pattern will be inhibited, enabling an uncommitted node to successfully compete to represent this pattern. On the other hand, if the pattern, to which this set of coactive nodes are responding, is just due to the coactivation of independent input patterns then the weights will return towards zero on subsequent presentations of these patterns in isolation.

For programming convenience we allow a single synapse to take either excitatory or inhibitory weight values. In a more biologically plausible implementation two separate sets of afferent connections could be used: the excitatory ones being trained using equation 2, and the inhibitory ones being trained using a rule similar to equation 3. Note that the simplification we have used, that employs one set of weights to depict both excitatory and inhibitory synapses, does not result in any modification to the activation function (equation 1): inhibitory afferents can themselves be inhibited and the use of identical lateral weights is unlikely to cause lateral facilitation due to the $\max$ operation that is applied.

3 Results

3.1 Overlap

Consider a simple problem of representing two overlapping patterns: ‘ab’ and ‘abc’. A network consisting of two nodes receiving input from three sources (labelled ‘a’, ‘b’ and ‘c’) should be sufficient. Because these input patterns overlap, when the pattern ‘ab’ is presented the node representing ‘abc’ will be partially activated, while when the pattern ‘abc’ is presented the node representing ‘ab’ will be fully activated. Hence, when the synaptic weights have certain values both nodes will respond with equal strength to the same pattern. For example, when the weights are all equal, both nodes will respond to pattern ‘ab’ with equal strength (Marshall, 1995). Similarly, when the total synaptic weight *from* each input is normalized (‘post-synaptic normalization’) both nodes will respond equally to pattern ‘ab’ (Marshall, 1995). When the total synaptic weight *to* each node is normalized (‘pre-synaptic normalization’) both nodes will respond to pattern ‘abc’ with equal output (Marshall, 1995). Under all these conditions the response fails to distinguish between distinct input patterns and competition via post-integration lateral inhibition can do no better than choosing a node at random.

Several solutions to this problem have been suggested. Some require adjusting the activations using a function of the total synaptic weight received by the node (*i.e.*, using the Webber Law (Marshall, 1995) or a masking field (Cohen and Grossberg, 1987; Marshall, 1995)). These solutions scale badly with the number of overlapping inputs, and do not work when (as is common practice) the total synaptic weight to each node is normalized. Other suggestions have involved tailoring the lateral weights to ensure the correct node wins the competition (Marshall, 1995; Marshall and Gupta, 1998). The most obvious, but most overlooked, solution would be to remove constraints placed on allowable values for synaptic weights (*e.g.*, normalization) which serve to prevent the input patterns being distinguished in weight space. It is simple to invent sets of weights which unambiguously classify the two overlapping patterns (*e.g.*, if both weights to the node representing ‘ab’ are 0.5 and each weight to the node representing ‘abc’ is 0.4 then each node responds most strongly to its preferred pattern and could then successfully inhibit the activation of the other node).

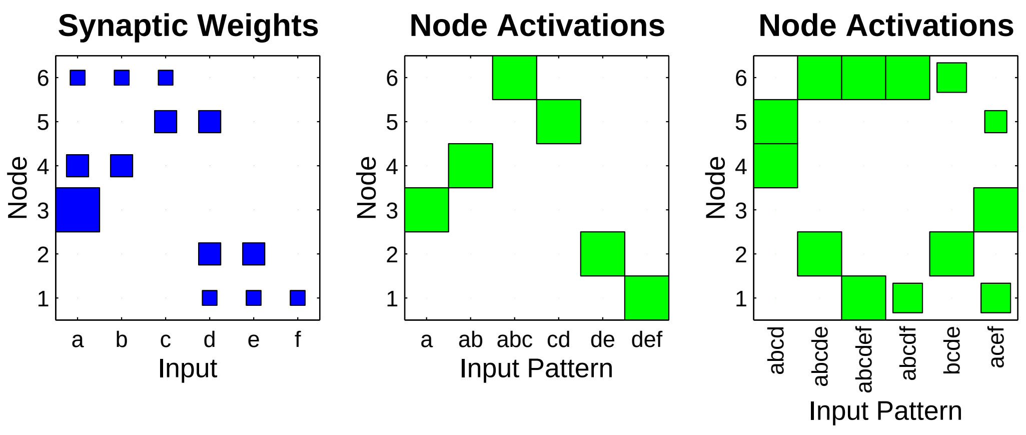
Using pre-integration lateral inhibition, overlapping patterns can be successfully distinguished and learnt (despite the use of pre-synaptic normalization). To demonstrate this we applied a six-node pre-integration lateral inhibition network to the more challenging problem of representing six overlapping patterns: ‘a’, ‘ab’, ‘abc’, ‘cd’, ‘de’, and ‘def’ (Marshall, 1995; Marshall and Gupta, 1998; Harpur and Prager, 1994; de A. Barreto and Araújo, 1998). The network was trained 25 times using different randomly generated sequences of the six input patterns. For each of these trials the network successfully learnt appropriate weights such that each pattern was represented by an individual node. A typical example of the weights learnt is shown in figure 2(a) and the response of the network to each of the training patterns is shown in figure 2(b). It can be seen that distinct, individual, nodes respond to each of the training patterns. For the majority of the 25 trials a solution was found within 55 training cycles (equivalent to approximately nine presentations of each of the six patterns). On the fastest trial the solution was found in 35 cycles and on the slowest trial the solution was found by 80 cycles. These results compare very favorably with a network (the EXIN network) that uses post-integration lateral inhibition and was specifically developed to solve these types of problem (Marshall, 1995; Marshall and Gupta, 1998; de A. Barreto and Araújo, 1998). Although Marshall (1995) gives no indication of how reliable EXIN is at finding a correct solution he does state that this network requires 3000 pattern presentations to find such a solution. Our results were generated using parameter values $\beta = 1$ and $\beta^{-} = 1$. The network learnt successfully with other parameter values, however, (as would be expected) reducing either learning rate resulted in slower learning (*e.g.*, with $\beta = 1$ and $\beta^{-} = \frac{1}{64}$ the majority of networks found a solution within 435 training cycles, the fastest trial took 245 cycles and the slowest 640 cycles).

The pre-integration lateral inhibition network also responds in an appropriate manner to the presentation of multiple, overlapping, patterns (even in case where only partial patterns are presented). Six examples are given in figure 2(c). These ‘parsings’ of novel input patterns, in terms of known patterns, are very similar to those obtained with the EXIN network (see figure 8 in Marshall, 1995).

3.2 Multiplicity

While it is sufficient in certain circumstances for a single node to represent the input (local coding) it is desirable in many other situations to have multiple nodes providing a factorial or distributed representation. If individual nodes act as detectors for independent features of the input, then any arbitrary pattern can be represented by having zero, one or multiple nodes active. A benchmark task of this kind is the bars problem (Földiák, 1990; Saund, 1995; Dayan and Zemel, 1995; Hinton et al., 1995; Harpur and Prager, 1996; Frey et al., 1997; Hinton and Ghahramani, 1997; Fyfe, 1997; Charles and Fyfe, 1998; Hochreiter and Schmidhuber, 1999; O’Reilly, 2001).

The standard mechanism of post-integration lateral inhibition can be modified to enable multiple nodes to be



(a) (b) (c)

Figure 2: Representing overlapping patterns. A network consisting of six nodes and six inputs (‘a’, ‘b’, ‘c’, ‘d’, ‘e’, and ‘f’) was trained using input patterns ‘a’, ‘ab’, ‘abc’, ‘cd’, ‘de’, and ‘def’. (a) The strength of synaptic weights between each node and each input are shown using squares which have sides of length proportional to the synaptic weight. (b) The average response (over 500 presentations) of each node to each of the training patterns is shown using squares which have sides of length proportional to the node activations. Each node responds exclusively to its preferred pattern. (c) The response of the network to multiple and partial patterns. Pattern ‘abcd’ causes the nodes representing ‘ab’ and ‘cd’ to be active simultaneously, despite the fact that this pattern overlaps strongly with pattern ‘abc’. Input ‘abcde’ is parsed as ‘abc’ together with ‘de’, and input ‘abcdef’ is parsed as ‘abc’ + ‘def’. Input ‘abcdf’ is parsed as ‘abc’ + two-thirds of ‘def’, hence the addition of ‘f’ to the pattern ‘abcd’ radically changes the representation that is generated. Input ‘bcde’ is parsed as two-thirds of ‘abc’ plus pattern ‘de’. Input ‘acef’ is parsed as ‘a’ + one half of ‘cd’ + two-thirds of pattern ‘def’.

coactive (Marshall, 1995; Földiák, 1990) by weakening the strength of the competition between those pairs of nodes that need to be coactive. Lateral weights need to be sufficiently strong to prevent multiple nodes representing the same patterns, but sufficiently weak to allow more than one node to be active if multiple stimuli are presented simultaneously. Obtaining the correct strength for the lateral weights either requires *a priori* knowledge of which nodes will be coactive or the ability to learn appropriate weights. However, the activity of the nodes in response to the presentation of each input pattern is uninformative as to whether the weights have found the correct compromise: when two nodes are simultaneously active it can indicate either that lateral weights are too weak and have allowed both units to respond to the same input, or that each node has responded correctly to the simultaneous presentation of distinct inputs; when a single node is active it can indicate either that the lateral weights are too strong to allow other nodes to respond to other inputs, or that only one input pattern is present. To overcome this problem it has been necessary to design learning rules specific to the particular task. For example, Földiák (1990) uses the following rule to modify lateral inhibitory weights: $\Delta q_{kj} = \beta \left(y_k y_j - p^2 \right)$ (where p is the probability of each input pattern occurring, assumed *a priori*). Such learning rules are severely restricted in the class of problems that they can successfully represent (*e.g.*, to tasks in which all input patterns occur with equal probability and where pairs of nodes are coactive with equal frequency (Marshall, 1995; Földiák, 1990)). Hence, post-integration lateral inhibition fails to provide factorial coding except in the exceptional circumstances where external knowledge is available to set appropriate lateral weights, or to define appropriate learning rules for the specific task.

Networks in which post-integration competition is performed using a selection mechanism can also be modified to allow multiple nodes to be simultaneously active (*e.g.*, k-winners-take-all). However, these networks also place restrictions on the types of task that can be successfully represented to those in which a pre-defined number of nodes need to be active in response to every pattern of stimuli.

In contrast, pre-integration lateral inhibition does not make use of *a priori* knowledge to learn factorial codes nor does it place restrictions on the number of active nodes, nor on the frequency with which nodes, or sets of nodes, are active. It can thus respond appropriately to any combination of input patterns. To demonstrate this we applied a network using pre-integration lateral inhibition to the bars problem, as defined by Földiák (1990). Input data consisted of an 8 by 8 pixel image in which each of the 16 possible (one-pixel wide) horizontal and vertical bars were active with probability $\frac{1}{8}$. Typical examples of input patterns are shown in figure 3(a). The network was trained 25 times using different randomly generated sequences of input patterns. For each of these trials the network successfully learnt appropriate weights such that each bar was represented by an individual node. A typical example of the weights learnt, at various stages during training, are shown in figure 3(b). The response of this network to randomly generated test patterns is shown in figure 3(c). It can be seen that a single node exclusively responds to each bar. Bars which overlap (*i.e.*, perpendicular bars) can inhibit each other, hence, in cases where perpendicular lines are present not all nodes are fully activated by the presence of their preferred input. However, nodes which represent active bars are still significantly more active than average.

After being trained for 250 cycles the network shown in figure 3 was tested with a further 100000 randomly generated patterns (learning rates were set to zero to prevent further training). The network was considered to successfully represent a test pattern if all nodes representing bars in the pattern, and only those nodes, had an activation greater than the average activation. The network failed to represent 13 patterns out of 100000. These 13 patterns are shown in figure 4 along with the network's response to them. In all cases the patterns contain 7 or 8 bars with the majority of these bars at the same orientation. The network successfully represents the majority of bars but nodes responding to the, one or two, perpendicular bars are less active than average.

In the majority of the 25 trials performed on this task, the network found a solution within 210 cycles. Performance ranged from 140 cycles for the fastest trail to 370 cycles for the slowest trial. Although Földiák (1990) developed a network which used post-integration lateral inhibition to tackle this problem he did not report either how reliable his network was at finding correct solutions nor how much training it required to do so. However, our results compare favorably with all neural network algorithms that have been applied to this problem, not just those that implement post-integration lateral inhibition. For example, Harpur and Prager (1996) report that training required 2000 pattern presentations[1], Hochreiter and Schmidhuber (1999) report that training required 5000 passes through a training set of 500 patterns[2] and Dayan and Zemel (1995) report a success rate of only 69%[3].

The results presented above were obtained with learning rates set to $\beta = 1$ and $\beta^- = \frac{1}{64}$. The network was also tested over 25 trials using $\beta^- = 1$. Although this case also resulted in a correct solution being found (within 1000 cycles) in 100% of trials, it did significantly delay convergence to a solution in a small number of these trials. In the overlapping patterns task, discussed in the previous section, training data consisted of isolated patterns. Hence, whenever multiple nodes were coactive it indicated that a pattern was incorrectly represented and negative weights should be increased. In contrast, for this task training data consists of multiple patterns, and hence coactive nodes may also be due to multiple input patterns being present. A lower value of β^- thus prevents useful weight changes caused by incorrect representations being masked by spurious fluctuations due to multiple input patterns. Note however, that it is not necessary to know *a priori* whether or not the training data contains multiple patterns: the bars problem can be learnt with 100% success rate using $\beta^- = 1$, also the overlapping inputs problem can be learnt with 100% success rate using $\beta^- = \frac{1}{64}$. Figure 5(a) illustrates the effect of changing the learning rate. This figure shows the change, during learning, of the number of bars correctly represented by the network. For a bar to be considered correctly represented exactly one node in the network must have strong afferent weights from the pixels activated by that bar. Specifically, the sum of these weights must be at least twice the sum of the weights connected to pixels activated by any other bar. The data shown is averaged over 25 trials, and the error bars indicate the best and worst performance.

All the previous analysis has been performed on a network with 16 nodes. In general, it would not be known *a priori* how many nodes were required to form a representation of a particular task. Hence, the algorithm must also work when the number of nodes does not match the number of input features. A network containing 32 nodes was tested 25 times using different randomly generated sequences of input patterns. In all these trials the network found a solution to the bars problem. The slowest trial took 440 cycles to find a solution (slightly longer than the worst trial with the 16-node network). Excess nodes generally remained uncommitted to representing any pattern (*i.e.*, all weights remained equal to their initial values). Occasionally, an excess node did form a slight preference to a complex pattern containing multiple bars. A typical example of the weights learnt is shown in figure 6. The representations formed in networks with excess nodes remained stable. We also tested a version of the algorithm in which the number of nodes in the network was not fixed, but nodes (up to a maximum number, in this case 20) were added as required during training. This network began with two nodes. Whenever none of the nodes remained uncommitted a new, uncommitted, node was added to the network. In the majority of 25 random trials performed with this network it converged to a solution within 130 training cycles. In the slowest trial a solution was found in 290 training cycles. This 'constructive' version of the algorithm thus learns more quickly than a network with a fixed number of nodes. Figure 5(b) summarizes the results obtained for networks with different numbers of nodes.

[1]Harpur and Prager (1996) used a slightly modified version of the original problem such that bars were active with probability $\frac{1}{4}$.

[2]Hochreiter and Schmidhuber (1999) used a version in which a 5 by 5 pixel image was used and bars appeared with probability $\frac{1}{5}$.

[3]Dayan and Zemel (1995) used the same task as Hochreiter and Schmidhuber (1999) except that horizontal and vertical bars could not co-occur. Our network tested with data in which horizontal and vertical bars did not co-occur learnt a correct solution in 100% of 25 trials. In the majority of trials a solution was found within 195 cycles.

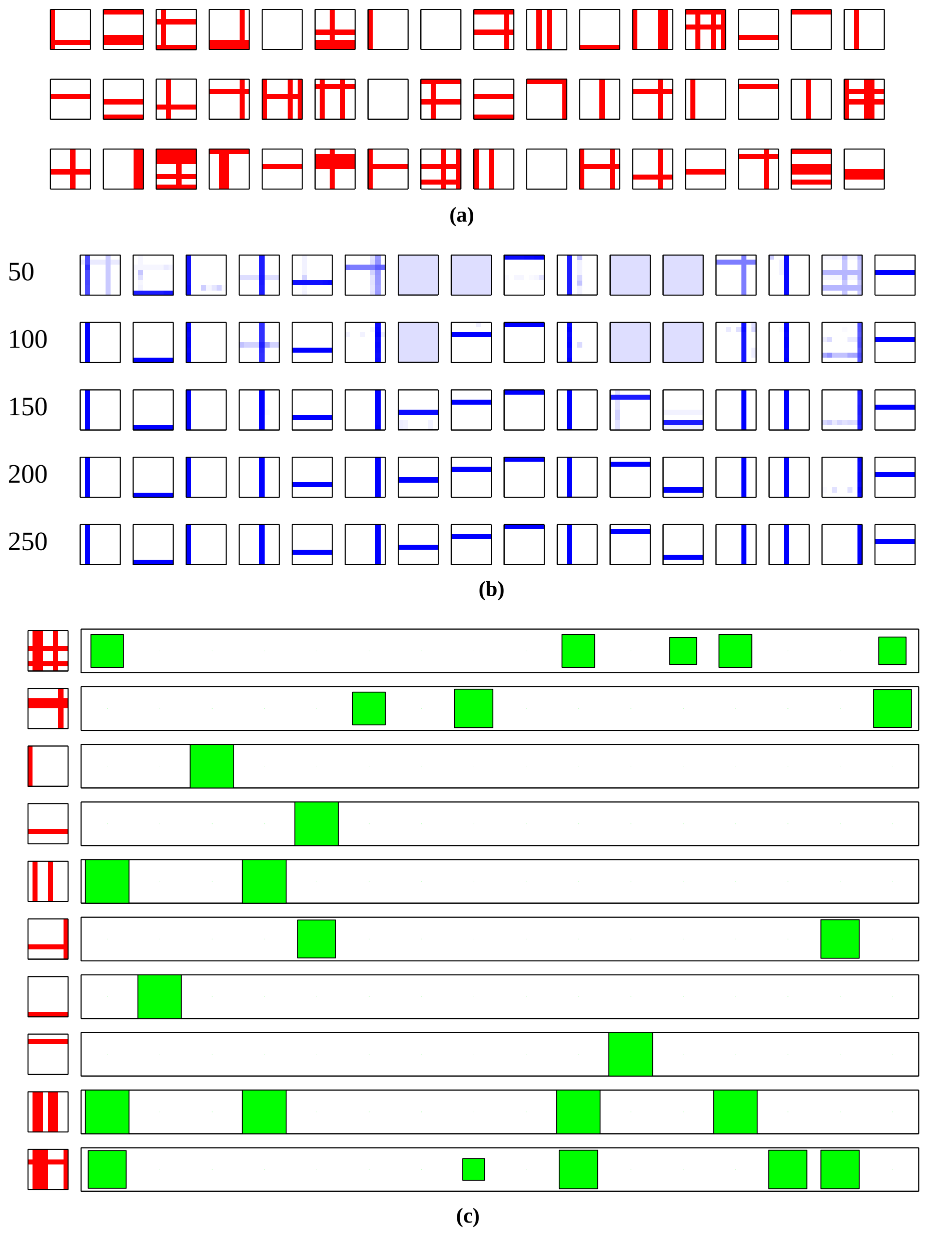


Figure 3: Representing multiple patterns: the bars problem. (a) Examples of typical input patterns used in the bars problem. Bars on an 8x8 grid are active with probability $\frac{1}{8}$. Dark pixels indicate active inputs. (b) The synaptic weights for the 16 nodes in a network after training with 50, 100, 150, 200 and 250 input patterns. The darkness of each pixel corresponds to the strength of that weight. (c) The response of the network after training on 250 input patterns. The left-most column shows the input pattern, with the remainder of each row showing the response of the network to this pattern. The size of the response of each node, in the same order as the weights are shown in (b), is represented by the size of each square.

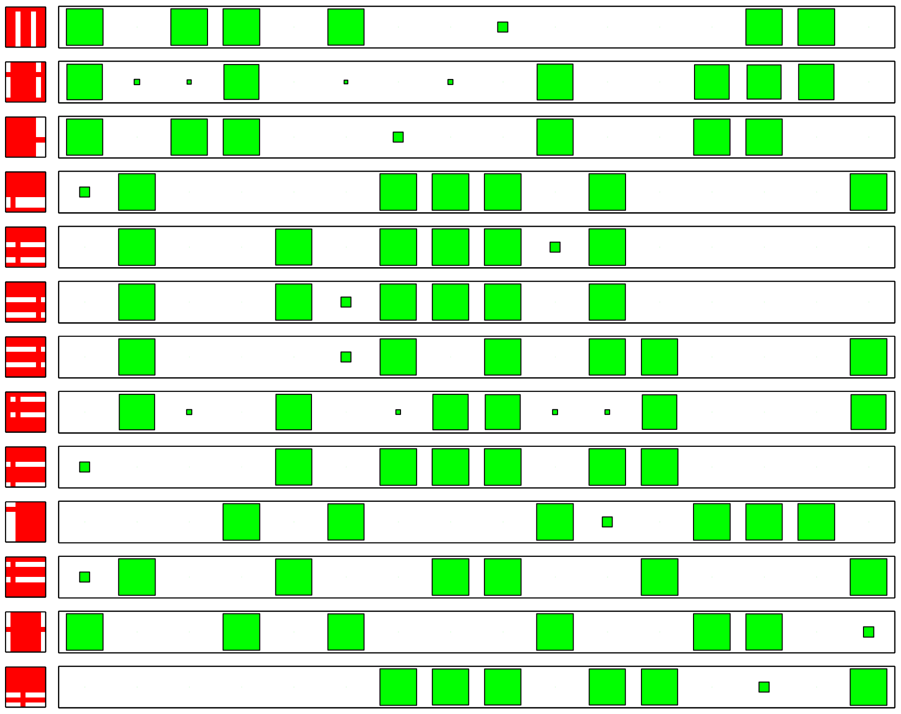


Figure 4: Failure cases for the bars problem. The network shown in figure 3 was tested with 100000 randomly generated patterns. All the 13 patterns for which it failed to respond correctly to are shown in the left-most column. The remainder of each row shows the response of the network to this pattern. The size of the response of each node, in the same order as the weights are shown in figure 3(b), is represented by the size of each square.

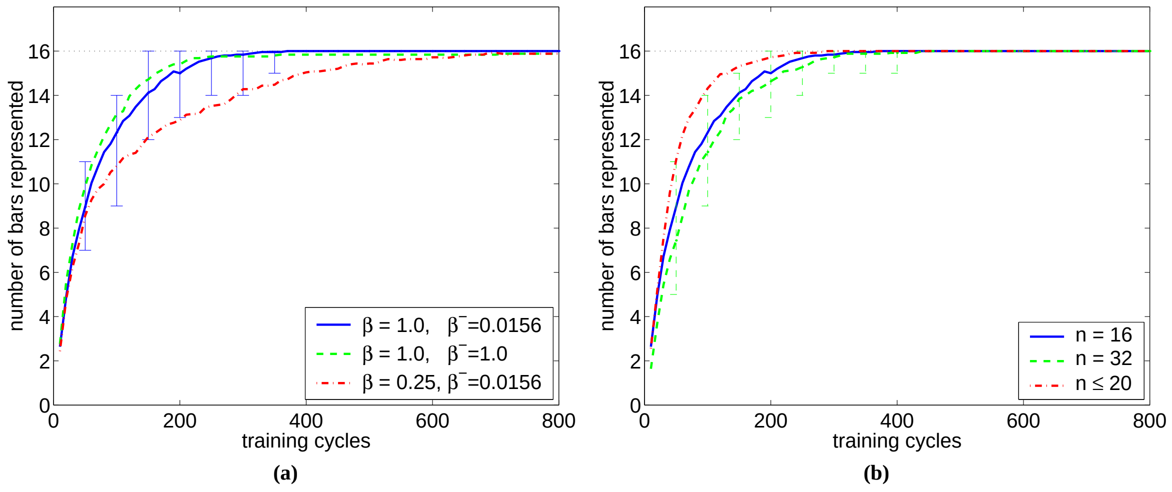


Figure 5: Effect of parameter changes in solving the bars problem. The change, during training, in the number of bars correctly represented by exactly one node in the network. Lines show the mean for 25 trials with different randomly generated sequences of input patterns. Error bars show the best and worst performance over all trials. (a) The effects of varying the learning rates. (b) The effects of varying the number of nodes in the network.

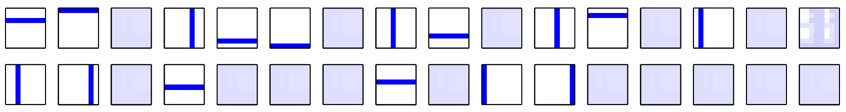


Figure 6: Typical receptive fields learnt by a 32-node network applied to the bars problem. Each square shows the synaptic weights for a node after training for 500 cycles. The darkness of each pixel corresponds to the strength of that weight.

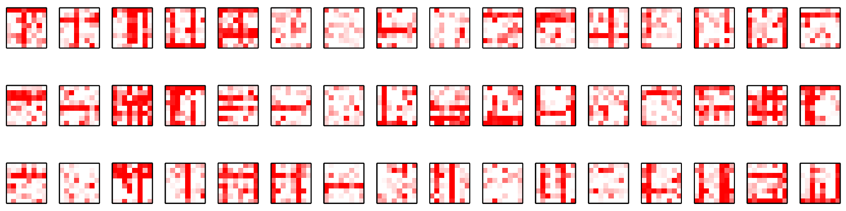
All the above experiments have used noise free stimuli. Such perfect input data is unrealistic for real-world problems. The performance of the pre-integration lateral inhibition network was thus tested on the bars problem, as described above, but with zero-mean Gaussian noise added independently to each pixel value (pixel values were clipped at zero and one). Two factors result in significantly slower learning when noisy data is used. Firstly, more examples of input patterns are needed to allow the algorithm to distinguish signal from noise. Secondly, a lower learning rate is needed in order to smooth out weight fluctuations caused by the noise. It was also found necessary to use a network in which there were excess nodes, these nodes learnt random looking weights. A typical example of the weights learnt are shown in figure 7(b). A 20-node network, with $\beta = 0.25$ and $\beta^{-} = \frac{1}{64}$, was trained 25 times with different randomly generated sequences of 4000 input patterns. Figure 8 summarizes the results obtained using data corrupted with vary degrees of noise. As the level of noise rises, the network is less reliable at finding a solution, and takes longer to do so. Although for higher levels of noise, the network did not find a solution, within 4000 training cycles, on every trial the performance is impressive considering the difficultly in perceiving the bars under this level of noise (see figure 7(a)). For comparison, Charles and Fyfe (1998) report on an algorithm with 100% success rate when the variance of the noise was 0.25 and 70% success rate when the variance was 1.0. However, their network was trained for 100000 cycles. Hinton and Ghahramani (1997) report that training required 10 passes through a set of 1000 images, but do not report the variance of the noise used, nor the reliability with which a solution was found[4].

4 Discussion

The above examples have shown that pre-integration lateral inhibition is capable of generating appropriate representations based on the 'knowledge' stored in the synaptic weights of a neural network. Specifically, it can represent overlapping patterns and it can respond to partial patterns such that the response is proportional to how well that input matches the stored pattern. It is capable of generating a local encoding of individual input patterns as well as responding simultaneously to multiple patterns, when they are present, in order to generate a factorial or distributed encoding. Importantly, such a network is capable of efficiently learning such representations using a simple, unsupervised, learning algorithm. In contrast, a network using post-integration lateral inhibition can learn to represent mutually exclusive input patterns using winner-take-all type competition, but it has not been shown how it can learn factorial codes except in cases where the input data meets very restrictive criteria. With pre-integration lateral inhibition learning is reliable and fast. Calculating node activations is tractable, requiring a polynomial-time algorithm, but is slightly slower than for post-integration lateral inhibition (computational complexity increases from $O(mn + n^2)$ for post-integration lateral inhibition to $O(mn^2)$ for pre-integration lateral inhibition).

We believe that the algorithm we have presented has sufficient computational qualities to make it interesting as an artificial, self-organizing, neural network architecture. However, we also suggest that it has potential application as a model of cortical information processing. Inhibitory synapses contacting cortical pyramidal cells are concentrated on the soma and axon initial segment (Somogyi and Martin, 1985; Mountcastle, 1998) where they can be equally effective at inhibiting responses to excitatory inputs stimulating any part of the dendritic tree (Spruston et al., 1999). Such synapses could thus generate post-integration inhibition. However, inhibitory synapses also contact the dendrites of cortical pyramidal cells (Kim et al., 1995; Rockland, 1998) and certain classes of interneuron (*e.g.*, double bouquet cells) specifically target dendritic spines and shafts (Mountcastle, 1998; Tamas et al., 1997). Such contacts would have relatively little impact on excitatory inputs more proximal to the cell body or on the action of synapses on other branches of the dendritic tree. Thus these synapses do not appear to contribute to post-integration inhibition. However, such synapses are likely to have strong inhibitory effects on inputs within the same dendritic branch that are more distal to the site of inhibition (Spruston et al., 1999; Borg-Graham et al., 1998; Koch et al., 1983; Rall, 1964; Segev, 1995; Koch and Segev, 2000). Hence, they could potentially selec-

[4]Hinton and Ghahramani (1997) used a 6 by 6 image, bars appeared with probability 0.3, and horizontal and vertical bars could not co-occur.


(a)

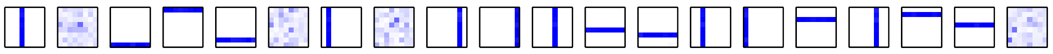
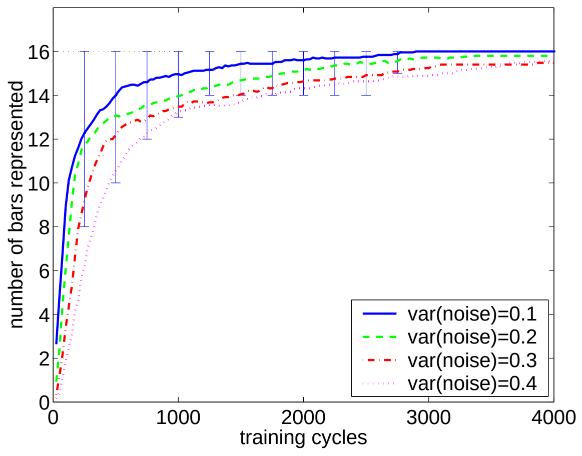

(b)

Figure 7: The noisy bars problem. (a) Examples of input patterns used in the noisy bars problem. The bars data is corrupted with zero-mean Gaussian noise, with variance 0.3, added independently to each pixel value (pixel values were clipped at zero and one). The darkness of each pixels corresponds to the strength of activation of that input.(b) Typical receptive fields learnt by a 20-node network applied to the noisy bars problem (using noise with variance 0.3). Each square shows the synaptic weights for a node after training for 4000 cycles. The darkness of each pixel corresponds to the strength of that weight.



(a)

Variance of noise	Reliability (%)	Speed (cycles)
0.1	100	1125
0.2	92	1900
0.3	76	2700
0.4	60	3550

(b)

Figure 8: Effect of changing levels of noise in solving the bars problem. The change, during training, in the number of bars correctly represented by exactly one node in the network. Lines show the mean for 25 trials with different randomly generated sequences of input patterns. Error bars show the best and worst performance over all trials. (a) The effects of varying the variance of the noise corrupting the input data. (b) Table showing reliability (the percentage of trials for which a solution was found by the end of 4000 training cycles), and speed (the number of training cycles required for the majority of trials to have reached a solution). In trials that failed to find a solution the network represented most of the bars (usually 15), hence the average number of bars represented, as shown in (a), is high even when the reliability is low.

tively inhibit specific groups of excitatory inputs. Related synapses cluster together within the dendritic tree so that local operations are performed by multiple, functionally distinct, dendritic subunits before integration at the soma (Koch and Segev, 2000; Segev and Rall, 1998; Segev, 1995; Häusser et al., 2000; Häusser, 2001; Mel, 1994, 1999). Dendritic inhibition could thus act to 'block' the output from individual functional compartments. If, for modeling convenience, all the synapses contributing to a dendritic compartment are grouped together as a single input then our algorithm can be viewed as a model of dendritic inhibition.

The idea embodied in our model is that pyramidal cells inhibit the activity of dendritic compartments in other pyramidal cells within the same cortical area. This claim is anatomically plausible since it has been shown that cortical pyramidal cells innervate inhibitory cell types which in turn form synapses on the dendrites of pyramidal cells (Buhl et al., 1997; Tamas et al., 1997). Our model is also supported by recent physiological data. Wang et al. (2000) report that blockade of GABAergic (inhibitory) synapses alters the selectivity of cells in monkey inferotemporal cortex (area TE). Application of bicuculline methiodide (a GABA$_A$ receptor antagonist) results in increased responses to certain image features but not others. "These results suggest that a substantial fraction of excitatory inputs to TE neurons normally are not expressed because of GABAergic inhibition . . . the effects of bicuculline on these neurons resulted from specific disinhibition of GABAergic synapses on the excitatory inputs of particular stimulus features, rather than removal of nonspecific inhibition" (Wang et al., 2000). This data is consistent with our model of dendritic inhibition but is difficult to account for with a model that uses only post-integration inhibition.

As with models of post-integration lateral inhibition we have simplified reality by assuming that the role of inhibitory cells can be approximated by direct inhibitory weights from excitatory cells. A more biologically accurate model might include a separate inhibitory cell population. One potential implementation would allocate one inhibitory neuron to each input or dendritic compartment (i). This inhibitory cell would receive weighted connections from each excitatory node. It would perform a $\max$ operation over these inputs and apply the resulting inhibition equally to input i of every excitatory node. Using this scheme each input (or dendritic compartment) would be inhibited by a single synapse. In sub-cortical structures, where axo-axonal, terminal-to-terminal, synapses are common (Brown, 1991; Kandel et al., 1995), our model might have an even more direct biological implementation since an excitatory neuron can directly inhibit the input to another neuron via an excitatory synapse on the axon terminal of the input ('pre-synaptic inhibition') (Bowsher, 1970; Shepherd, 1990; Kandel et al., 1995).

Acknowledgements

This work was funded by MRC Research Fellowship number G81/512.

References

Borg-Graham, L. T., Monier, C., and Fregnac, Y. (1998). Visual input evokes transient and strong shunting inhibition in visual cortical neurons. *Nature*, 393(6683):369–73.

Bowsher, D. (1970). *Introduction to the Anatomy and Physiology of the Nervous System*. Blackwell, Oxford.

Brown, A. G. (1991). *Nerve Cells and Nervous Systems: An Introduction to Neuroscience*. Springer-Verlag, London.

Buhl, E. H., Tamas, G., Szilagyi, T., Stricker, C., Paulsen, O., and Somogyi, P. (1997). Effect, number and location of synapses made by single pyramidal cells onto aspiny interneurones of cat visual cortex. *The Journal of Physiology,*, 500(3):689–713.

Charles, D. and Fyfe, C. (1998). Modelling multiple cause structure using rectification constraints. *Network: Computation in Neural Systems*, 9(2):167–82.

Cohen, M. A. and Grossberg, S. (1987). Masking fields: a massively parallel neural architecture for learning, recognizing, and predicting multiple groupings of patterned data. *Applied Optics*, 26:1866–91.

Dayan, P. and Zemel, R. S. (1995). Competition and multiple cause models. *Neural Computation*, 7:565–79.

de A. Barreto, G. and Araújo, A. F. R. (1998). The role of excitatory and inhibitory learning in EXIN networks. In *Proceedings of the IEEE World Congress on Computational Intellignece*, pages 2378–83, New York. IEEE Press.

Földiák, P. (1989). Adaptive network for optimal linear feature extraction. In *Proceedings of the IEEE/INNS International Joint Conference on Neural Networks*, volume 1, pages 401–5, New York. IEEE Press.

Földiák, P. (1990). Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics*, 64:165–70.

Földiák, P. (1991). Learning invariance from transformation sequences. *Neural Computation*, 3:194–200.

Frey, B. J., Dayan, P., and Hinton, G. E. (1997). A simple algorithm that discovers efficient perceptual codes.

In Jenkin, M. and Harris, L. R., editors, *Computational and Psychophysical Mechanisms of Visual Coding*. Cambridge University Press, Cambridge.

Fyfe, C. (1997). A neural net for PCA and beyond. *Neural Processing Letters*, 6(1-2):33–41.

Grossberg, S. (1976). Adaptive pattern classification and universal recoding, I: parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23:121–134.

Grossberg, S. (1978). A theory of human memory: self-organisation and performance of sensory-motor codes, maps, and plans. In Rosen, R. and Snell, F., editors, *Progress in Theoretical Biology*, volume 5, pages 233–374, London. Academic Press.

Grossberg, S. (1987). Competitive learning: from interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63.

Grzywacz, N. M. and Burgi, P.-Y. (1998). Towards a biophysically plausible bidirectional Hebbian rule. *Neural Computation*, 10:499–520.

Harpur, G. and Prager, R. (1996). Development of low entropy coding in a recurrent network. *Network: Computation in Neural Systems*, 7(2):277–84.

Harpur, G. F. and Prager, R. W. (1994). A fast method for activating competitive self-organising neural networks. In *Proceedings of the International Symposium on Artificial Neural Networks*, pages 412–8.

Häusser, M. (2001). Synaptic function: dendritic democracy. *Current Biology*, 11:R10–2.

Häusser, M., Spruston, N., and Stuart, G. J. (2000). Diversity and dynamics of dendritic signalling. *Science*, 290(5492):739–44.

Hertz, J., Krogh, A., and Palmer, R. G. (1991). *Introduction to the Theory of Neural Computation*. Addison-Wesley, Redwood City, California.

Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. (1995). The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–61.

Hinton, G. E. and Ghahramani, Z. (1997). Generative models for discovering sparse distributed representations. *Philosophical Transactions of the Royal Society of London. Series B*, 352:1177–90.

Hochreiter, S. and Schmidhuber, J. (1999). Feature extraction through LOCOCODE. *Neural Computation*, 11:679–714.

Kandel, E. R., Schwartz, J. H., and Jessell, T. M., editors (1995). *Essentials of Neural Science and Behavior*. Appleton & Lange, London.

Kim, H. G., Beierlein, M., and Connors, B. W. (1995). Inhibitory control of excitable dendrites in neocortex. *Journal of Neurophysiology*, 74(4):1810–4.

Koch, C., Poggio, T., and Torre, V. (1983). Nonlinear interactions in a dendritic tree: localization, timing, and role in information processing. *Proceedings of the National Academy of Science USA*, 80(9):2799–802.

Koch, C. and Segev, I. (2000). The role of single neurons in information processing. *Nature Neuroscience*, 3(supplement):1171–7.

Kohonen, T. (1997). *Self-Organizing Maps*. Springer, Berlin; London.

Marshall, J. A. (1995). Adaptive perceptual pattern recognition by self-organizing neural networks: context, uncertainty, multiplicity, and scale. *Neural Networks*, 8(3):335–62.

Marshall, J. A. and Gupta, V. S. (1998). Generalization and exclusive allocation of credit in unsupervised category learning. *Network: Computation in Neural Systems*, 9(2):279–302.

Mel, B. W. (1994). Information processing in dendritic trees. *Neural Computation*, 6:1031–85.

Mel, B. W. (1999). Why have dendrites? A computational perspective. In Stuart, G., Spruston, N., and Häusser, M., editors, *Dendrites*, chapter 11, pages 271–89. Oxford University Press, Oxford.

Mountcastle, V. B. (1998). *Perceptual Neuroscience: The Cerebral Cortex*. Harvard University Press, Cambridge, Massachusetts; London.

Nigrin, A. (1993). *Neural Networks for Pattern Recognition*. MIT Press, Cambridge, Massachusetts; London.

Oja, E. (1989). Neural networks, principle components, and subspaces. *International Journal of Neural Systems*, 1:61–8.

O'Reilly, R. C. (1998). Six principles for biologically based computational models of cortical cognition. *Trends in Cognitive Sciences*, 2(11):455–62.

O'Reilly, R. C. (2001). Generalization in interactive networks: The benefits of inhibitory competition and Hebbian learning. *Neural Computation*, 13(6):1199–1242.

Rall, W. (1964). Theoretical significance of dendritic trees for neuronal input-output relations. In Reiss, R. F., editor, *Neural Theory and Modeling*, pages 73–97. Stanford University Press, Stanford, California.

Ritter, H., Martinetz, T., and Schulten, K. (1992). *Neural Computation and Self-Organizing Maps. An Introduction*. Addison-Wesley, Reading, Massachusetts.

Rockland, K. S. (1998). Complex microstructures of sensory cortical connections. *Current Opinion in Neurobiology*, 8:545–51.

Rumelhart, D. E. and Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, 9:75–112.

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–73.

Saund, E. (1995). A multiple cause mixture model for unsupervised learning. *Neural Computation*, 7(1):51–71.

Segev, I. (1995). Dendritic processing. In Arbib, M. A., editor, *The Handbook of Brain Theory and Neural Networks*, pages 282–9. MIT Press, Cambridge, Massachusetts; London.

Segev, I. and Rall, W. (1998). Excitable dendrites and spines: earlier theoretical insights elucidate recent direct observations. *Trends in Neurosciences*, 21(11):453–60.

Sejnowski, T. J. (1977). Storing covariance with nonlinearly interacting neurons. *Journal of Mathematical Biology*, 4:303–21.

Shepherd, G. M., editor (1990). *The Synaptic Organization of the Brain*. Oxford University Press, Oxford.

Shepherd, G. M. and Brayton, R. K. (1987). Logic operations are properties of computer-simulated interactions between excitable dendritic spines. *Neuroscience*, 21:151–66.

Sirosh, J. and Miikkulainen, R. (1994). Cooperative self-organization of afferent and lateral connections in cortical maps. *Biological Cybernetics*, 71:66–78.

Somogyi, P. and Martin, K. A. C. (1985). Cortical circuitry underlying inhibitory processes in cat area 17. In Rose, D. and Dobson, V. G., editors, *Models of the Visual Cortex*, chapter 54. Wiley, Chichester.

Spruston, N., Stuart, G., and Häusser, M. (1999). Dendritic integration. In Stuart, G., Spruston, N., and Häusser, M., editors, *Dendrites*, chapter 10, pages 231–271. Oxford University Press, Oxford.

Swindale, N. V. (1996). The development of topography in the visual cortex: a review of models. *Network: Computation in Neural Systems*, 7(2):161–247.

Tamas, G., Buhl, E. H., and Somogyi, P. (1997). Fast IPSPs elicited via multiple synaptic release sites by different types of GABAergic neurone in the cat visual cortex. *The Journal of Physiology,*, 500(3):715–38.

von der Malsburg, C. (1973). Self-organisation of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85–100.

Wallis, G. (1996). Using spatio-temporal correlations to learn invariant object recognition. *Neural Networks*, 9(9):1513–9.

Wang, Y., Fujita, I., and Murayama, Y. (2000). Neuronal mechanisms of selectivity for object features revealed by blocking inhibition in inferotemporal cortex. *Nature Neuroscience*, 3(8):807–13.