Neural Networks 21 (2008) 888–903

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Learning transform invariant object recognition in the visual system with multiple stimuli present during training

S.M. Stringer, E.T. Rolls *

Oxford University, Centre for Computational Neuroscience, Department of Experimental Psychology, South Parks Road, Oxford OX1 3UD, England, United Kingdom

ARTICLE INFO

Article history:
Received 19 September 2006
Accepted 7 November 2007

Keywords:
Object recognition
Inferior temporal cortex
Competitive neural networks
Continuous transformation learning
Trace learning

ABSTRACT

Over successive stages, the visual system develops neurons that respond with view, size and position invariance to objects or faces. A number of computational models have been developed to explain how transform-invariant cells could develop in the visual system. However, a major limitation of computer modelling studies to date has been that the visual stimuli are typically presented one at a time to the network during training. In this paper, we investigate how vision models may self-organize when multiple stimuli are presented together within each visual image during training. We show that as the number of independent stimuli grows large enough, standard competitive neural networks can suddenly switch from learning representations of the multi-stimulus input patterns to representing the individual stimuli. Furthermore, the competitive networks can learn transform (e.g. position or view) invariant representations of the individual stimuli if the network is presented with input patterns containing multiple transforming stimuli during training. Finally, we extend these results to a multi-layer hierarchical network model (VisNet) of the ventral visual system. The network is trained on input images containing multiple rotating 3D objects. We show that the network is able to develop view-invariant representations of the individual objects.

1. Introduction

An important problem in understanding natural vision is how the brain can build invariant representations of individual objects even when multiple objects are present in a scene. What mechanisms enable the learning to proceed without the different objects interacting with each other to interfere with the learning of individual object representations? In this paper we describe and analyze an approach to this which relies on the statistics of natural environments. The approach takes account of the statistics that any object can be present with any one of a number of other objects or backgrounds during learning, with some statistical independence of any one object from other objects in the scene.

Over successive stages, the visual system develops neurons that respond with view, size and position (translation) invariance to objects or faces (Desimone, 1991; Perrett & Oram, 1993; Rolls, 1992, 2000; Rolls & Deco, 2002; Tanaka, Saito, Fukada, & Moriya, 1991). For example, it has been shown that the inferior temporal visual cortex has neurons that respond to faces and objects with translation (Ito, Tamura, Fujita, & Tanaka, 1995; Kobotake & Tanaka, 1994; Op de Beeck & Vogels, 2000; Tovee, Rolls, & Azzopardi, 1994), size (Ito et al., 1995; Rolls & Baylis, 1986), and view (Booth & Rolls, 1998; Hasselmo, Rolls, Baylis, & Nalwa, 1989) invariance. Such invariant representations, once learned by viewing a number of the transforms of the object, are then useful in the visual system for allowing one-trial learning of, for example, the stimulus–reward association of the object, to generalize to other transforms of the same object (Rolls, 2005; Rolls & Deco, 2002).

A number of computational models have been developed to explain how transform-invariant cells could develop in the visual system (Fukushima, 1980; Riesenhuber & Poggio, 1999; Rolls, 2008; Wallis & Rolls, 1997). Two major theories which have sought to explain how transform-invariant representations could arise through unsupervised training with real-world visual input are *trace learning* (Földiák, 1991; Rolls & Milward, 2000; Wallis & Rolls, 1997), and *Continuous Transformation (CT) learning* (Perry, Rolls, & Stringer, 2006; Stringer, Perry, Rolls, & Proske, 2006). Trace learning relies on the temporal continuity of visual objects in the real world (as does slow feature analysis (Wiskott & Sejnowski, 2002)), while in contrast, CT learning relies on spatial continuity.

In most previous studies, however, of invariance learning in hierarchical networks that model the ventral visual stream, only one stimulus is presented at a time during training (Rolls & Milward, 2000; Rolls & Stringer, 2006; Stringer et al., 2006; Wallis & Rolls, 1997). In this paper we investigate whether, and if so how, models of this type can self-organize during training when multiple stimuli are presented together within each visual image.

* Corresponding author. Tel.: +44 1865 271419; fax: +44 1865 310447.
E-mail addresses: simon.stringer@psy.ox.ac.uk (S.M. Stringer), Edmund.Rolls@oxcns.org (E.T. Rolls).
URLs: http://www.oftnai.org (S.M. Stringer), http://www.oxcns.org (E.T. Rolls).

0893-6080/$ – see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2007.11.004

In Section 3 we show how a standard 1-layer competitive network responds when trained on input patterns containing multiple, e.g. pairs of, independent stimuli. As the number of stimuli N increases, the number of possible input patterns which are composed of pairs of stimuli $N(N-1)/2$ grows quadratically. For small numbers of stimuli N, the output neurons still represent the paired-stimulus input patterns. However, for large enough N, the output neurons begin to learn to respond to the individual stimuli instead of the multi-stimulus input patterns used during training. In this way, we show that a standard competitive network may suddenly switch from learning representations of the multi-stimulus input patterns to representing the individual stimuli as the number of independent stimuli grows large enough.

In Section 4 we continue to investigate how a 1-layer competitive network may learn to process input patterns containing multiple stimuli. However, we now extend the simulations by allowing the independent stimuli to transform, e.g. translate across the input space, during training. We show that, even when the network is trained on input patterns containing pairs of transforming stimuli, a standard 1-layer competitive network is able to learn invariant representations of the individual stimuli.

In Section 5 we extend these results to a full 4-layer hierarchical feedforward network model (VisNet) of the ventral visual processing stream. The visual input stimuli are rotating 3D objects created using the OpenGL 3D image generation software. We train the network on input images containing multiple rotating objects. We demonstrate that the network is able to develop view-invariant representations of the individual objects.

2. A hypothesis on how learning can occur about single objects even when multiple objects are present

We consider how learning about individual objects can occur even when a number of objects are present. Consider a situation that might occur in the real world in which an individual object is present, but is accompanied by one other object from a set of other objects. A possible series of trials might have (where each number refers to an individual object) all possible pairs of objects. For the case of $N = 4$ stimuli, there are a total of 6 paired-stimulus training patterns used on different trials as follows: $1 + 2$, $1 + 3$, $1 + 4$, $2 + 3$, $2 + 4$ and $3 + 4$. We approach the categorization that would occur using a competitive network, as this type of network performs categorization, is biologically plausible, and is the prototypical network involved in hierarchical models of object recognition (Hertz, Krogh, & Palmer, 1991; Rolls & Deco, 2002). We are interested in whether the network forms representations of the object pairs actually shown during training, or whether it can form, as desired for this application, separate representations of each individual object.

Consider the associations between the activity of the input neurons for each stimulus. For stimulus 1, the set of neurons that provide the representation of stimulus 1 have high correlations with each other. Moreover, these are higher correlations than those between the neurons representing different stimuli. (This occurs because on different trials the neurons representing stimulus 1 are sometimes paired with those representing stimulus 2, sometimes with those representing stimulus 3, etc.) We can calculate the number of times that the neurons within a stimulus are active simultaneously during one training epoch in which all pairs are presented, and with N stimuli, this is $N - 1$. In contrast, the neurons from different stimuli are co-active on only one occasion in a training epoch. The ratio of the within-stimulus to between-stimulus co-occurrences is thus $N - 1 : 1$. This increases with the number of stimuli. Given that competitive networks effectively build categories based on correlations between sets of afferents to the network, we propose that as N increases, a stage will be reached where instead of representing the pairs of stimuli actually presented during training, the network will instead form representations of the individual stimuli. We examine whether this transition occurs in Section 3.

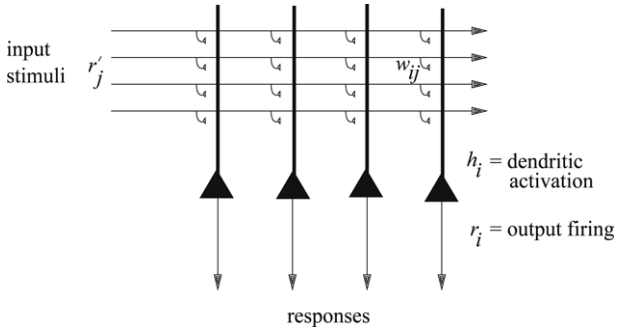

Fig. 1. The architecture of a competitive neural network. There is an input layer of cells with feedforward associatively modifiable synaptic connections onto an output layer of cells. At each timestep during learning, an input pattern is applied to the layer of input cells. Next, activity from the input layer is propagated through the feedforward connections to activate a winning set of cells in the output layer. Within the output layer there is feedback inhibition implemented by inhibitory neurons not shown in the figure, which ensures that only a small subset of output cells remains active. Next, the synaptic weights between the active input cells and the active output cells are strengthened. In this way, the output cells become associated with particular patterns of activity in the input layer.

3. Training a 1-layer competitive network with input patterns which contain multiple independent stimuli

In this section we show how the mechanism described in Section 2 operates in a 1-layer competitive network.

3.1. Model

The neural network architecture is shown in Fig. 1. The fully connected 1-layer model has the architecture of a standard competitive network (Hertz et al., 1991; Rolls & Deco, 2002). There is an input layer of cells with feedforward associatively modifiable synaptic connections onto an output layer of cells. At each timestep during learning, an input pattern is applied to the layer of input cells. Next, activity from the input layer is propagated through the feedforward connections to activate a set of cells in the output layer. Within the output layer there is competition implemented by feedback inhibition and the threshold nonlinearity of neurons. Next, the synaptic weights between the active input cells and the active output cells are strengthened by associative (Hebbian) learning. The output cells self-organize to represent and thus categorize different patterns of activity in the input layer.

The competitive network contained 100 input cells, 100 output cells, and was fully connected. Each simulation experiment used a set number N of independent stimuli. The stimulus patterns are represented by the activation of non-overlapping contiguous blocks of $100/N$ cells. Stimulus 1 is represented by cells 1 to $100/N$, stimulus 2 by the next block of $100/N$ cells, and so on up to N stimuli. For the case $N = 4$, Fig. 2 (left) shows the input representations of the 4 independent stimuli. (In this paper, by 'independent stimuli' we refer to stimuli of the type labelled as 1–4 in Fig. 2 left, and note that each such stimulus is paired with each of the other stimuli equally often during the training.) Fig. 2 (right) shows the 6 paired-stimulus input patterns that were used to train the 1-layer competitive network for the case $N = 4$, in which the training patterns are: $1 + 2$, $1 + 3$, $1 + 4$, $2 + 3$, $2 + 4$ and $3 + 4$.

During training, the agent is presented in turn with each of the possible $N(N-1)/2$ paired-stimulus input patterns. At the presentation of each training pattern, the input cells are set to fire

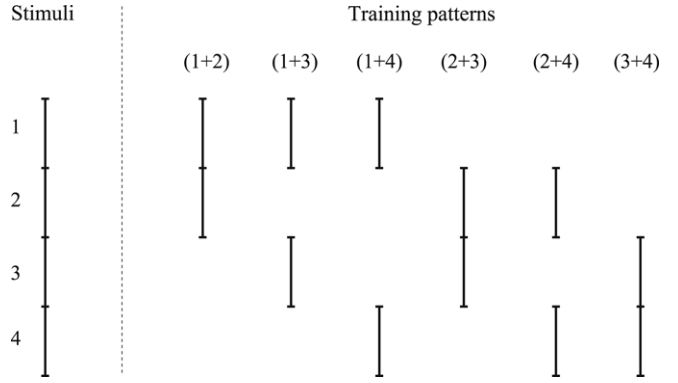

Fig. 2. The input representations of the independent stimuli and the paired-stimulus input patterns used to respectively test and train the 1-layer competitive network. The competitive network has 100 input cells arranged in a vertical line. Each simulation experiment uses a set number of independent stimuli N. The stimulus patterns are represented by the activation of non-overlapping contiguous blocks of $100/N$ cells. Stimulus 1 is represented by cells 1 to $100/N$, stimulus 2 by the next block of $100/N$ cells, and so on up to N stimuli. The case of $N = 4$ stimuli is represented in the figure. Also shown are the corresponding 6 paired-stimulus training patterns. For the case of $N = 4$ stimuli, there are a total of 6 paired-stimulus training patterns as follows: $1+2$, $1+3$, $1+4$, $2+3$, $2+4$ and $3+4$. The cells that are active in each pattern are shown as black.

according to the paired-stimulus activity training patterns shown in Fig. 2 (right). The activity from the input layer is then propagated through the feedforward synaptic connections to activate a set of cells in the output layer. (The synaptic weights from the input to output cells are initially set to random values from a uniform distribution in the range 0–1, and then normalized to a vector length of 1. This is standard in competitive networks, and ensures that some firing of output cells will be produced by the inputs, with each output cell likely to fire at a different rate for any one input stimulus.) The activations of the cells in the output layer are calculated according to

$$h_i = \sum_j w_{ij} r_j, \tag{1}$$

where h_i is the activation of output cell i, r_j is the firing rate of input cell j, and w_{ij} is the synaptic weight from input cell j to output cell i.

The activation h_i of each neuron was converted to the firing rate r_i of each neuron using a threshold linear activation, and adjusting the threshold to achieve a prescribed sparseness of the representation in the output cells. (This represents a process by which mutual inhibition between the output cells through inhibitory interneurons implements competition to ensure that there is only a small winning set of output cells left active.) The sparseness a of the representation that was prescribed was defined, by extending the binary notion of the proportion of neurons that are firing, as follows

$$a = \frac{\left(\sum_{i=1}^{M} r_i/M\right)^2}{\sum_{i=1}^{M} r_i^2/M}, \tag{2}$$

where r_i is the firing rate of the ith neuron in the set of M neurons (Rolls & Treves, 1990, 1998). In the simulations, the competition was achieved in an iterative cycle by feedback adjustment of the threshold for the firing of neurons until the desired sparseness was reached. The threshold was the same for all neurons. With the graded firing rates of the neurons, the result was that typically the proportion of neurons with non-zero firing rates after the competition was numerically somewhat larger than the sparseness value given as a parameter to the network.

Next, the synaptic weights between the active input cells and the active output cells are strengthened according to the associative Hebb learning rule

$$\delta w_{ij} = k r_i r_j \tag{3}$$

where δw_{ij} is the change of synaptic weight, and k is the learning rate constant which was set to 0.001 unless otherwise stated. To prevent the same few neurons from always winning the competition, the synaptic weight vectors are set to unit length after each learning update for each training pattern. To implement weight normalization the synaptic weights were rescaled to ensure that for each output cell i we have

$$\sqrt{\sum_j (w_{ij})^2} = 1, \tag{4}$$

where the sum is over all input cells j. Such a renormalization process may be achieved in biological systems through heterosynaptic long-term depression that depends on the existing value of the synaptic weight (Oja, 1982; Rolls & Deco, 2002; Rolls & Treves, 1998). (Heterosynaptic long-term depression in the brain was described by Levy (1985) and Levy and Desmond (1985); see Brown, Kairiss, and Keenan (1990)).

The presentation of all possible $N(N-1)/2$ stimulus-pair training patterns corresponded to one training epoch. For each experiment with fixed N, there were 10,000 training epochs to ensure convergence of the synaptic weights.

For each experiment, after training, the network was tested with N single-stimulus input patterns, each of which contained a different one of the independent stimuli as illustrated at the left of Fig. 2. During this testing, for each output cell we recorded how many of the N stimuli the cell responded to. The output neurons tended to be binary, that is to either be firing with a high rate or not at all for a given stimulus (as will be evident when Figs. 3 and 4 are described). An output cell was classed as responsive to a particular input stimulus if presentation of the input stimulus elicited a firing rate in the output cell which was greater than 50% of the maximal firing rate of any output cell to any stimulus.

3.2. Simulation results

The results described next showed that if $N \leq 5$ the network represented every one of the $N(N-1)/2$ paired-stimulus input patterns, and generally none of the N single-stimulus input patterns. If $N \geq 6$ the network represented generally none of the $N(N-1)/2$ paired-stimulus input patterns, and all of the N single-stimulus input patterns. These results are shown in Table 1. We remind the reader that the network was in all cases trained on the $N(N-1)/2$ paired-stimulus input patterns.

Table 1 summarizes the results obtained from simulation experiments, which show how the output cells of the 1-layer competitive network learn to represent the input data when different numbers of independent stimuli N are used to generate multi-stimulus training patterns. For the experimental results shown in Table 1, the population sparseness of the output firing rates a was always set to 0.05, and the average results obtained over 6 separate simulation experiments for a particular fixed value of N are shown. In all of the simulation experiments, after training, the output cells were found to be responsive to either 0 stimuli, a single stimulus, or 2 stimuli. No cells were ever found which responded to 3 or more stimuli. The standard errors of the means shown in Table 1 were in most cases less than 1. In all these simulations approximately equal numbers of cells learned to respond to each of the single stimuli for $N \geq 6$ or paired stimuli if $N \leq 5$ (as will be made evident in Fig. 4).

The ability of the network to learn to represent the independent component stimuli when N is large is quite robust with respect

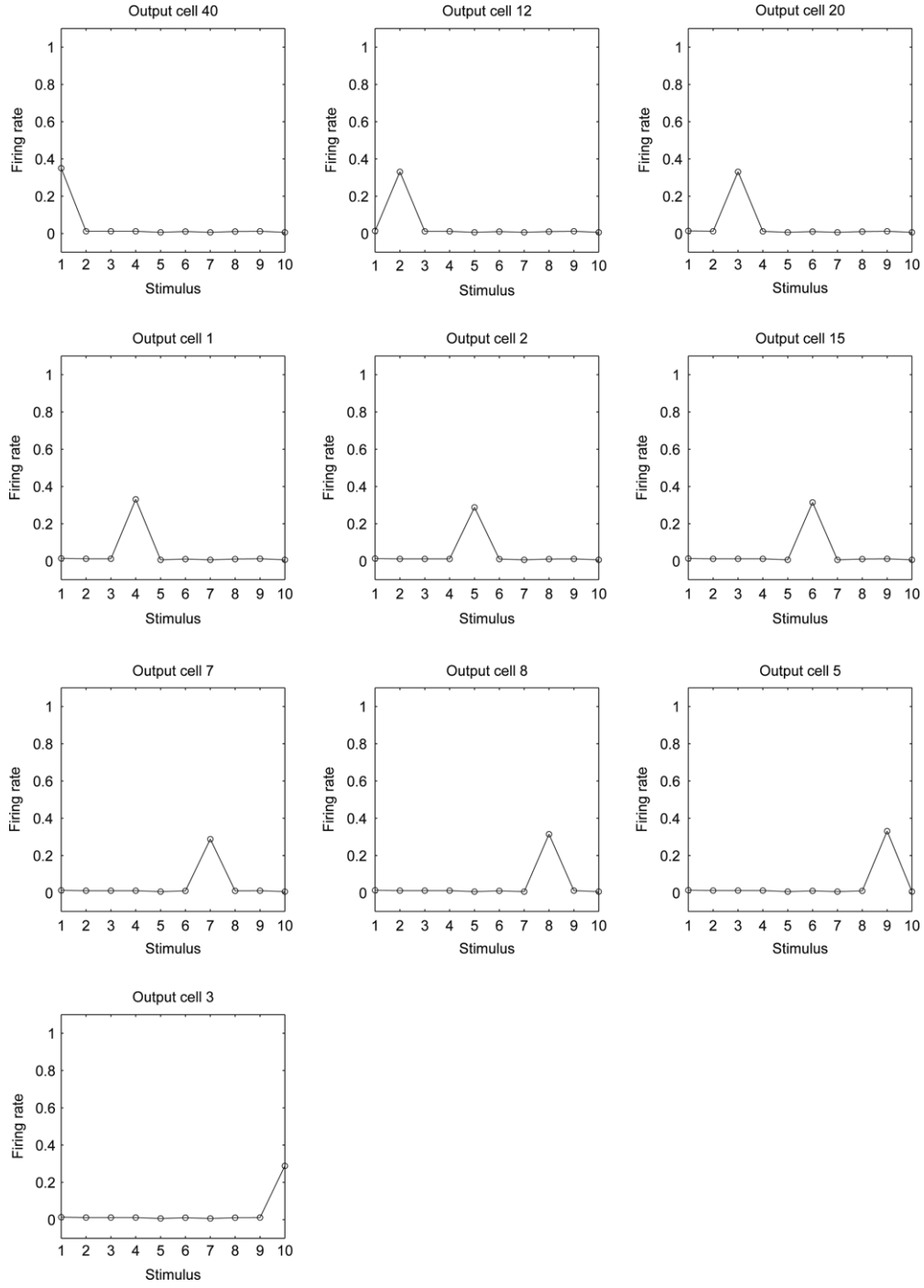

Fig. 3. The firing rate responses of a selection of 10 output cells to $N = 10$ independent stimuli, after the network has been trained on all possible stimulus-pair patterns constructed from the stimuli. The sparseness of the output firing rates was set to 0.2 during training and testing. For this test case, all 100 output cells learned to respond to a single stimulus, with disjoint subsets of approximately 10 output cells having learned to respond to each of the independent stimuli. This figure shows the responses of a selection of 10 output cells, each of which had learned to respond to a different one of the 10 stimuli.

to the sparseness of the output firing rates a, as shown by the results obtained for different values of the sparseness and $N = 10$ in Table 2. For the wide range of sparseness values tested, there were always mainly cells that responded only to a single independent stimulus, and rarely more than a few cells that learned to respond to the paired-stimulus training patterns. As before, the average values for 6 runs are shown, and the standard errors were small. With small values of the sparseness parameter a, only a few neurons were activated by a given stimulus, and most of the neurons therefore remained unallocated to any stimulus, retaining their initial random synaptic weights. With $a = 0.2$ approximately 20 of the 100 output cells were activated by each input stimulus-pair pattern during training. This corresponds to each single stimulus activating approximately 10 cells, and this is why as shown in Table 2 all 100 output cells became responsive to only a single stimulus after training, with a different set of 10 output cells associated with each of the 10 stimuli. Moreover, the number of output neurons is not the factor that determines the representations formed, for when the transition occurs to represent individual stimuli (see Table 1), the number of stimuli N equals 6, the number of stimulus pairs is 15, and the number of output neurons is 100.

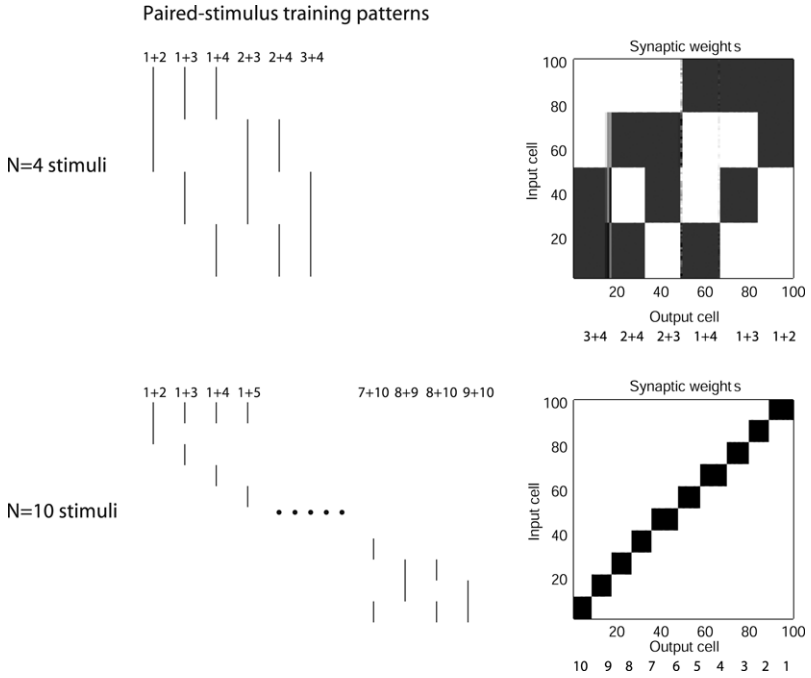

Fig. 4. The synaptic weights after the network has been trained with paired-stimulus input patterns constructed from different numbers of stimuli N. The top row shows results for the case $N = 4$ stimuli with $a = 0.14$. On the left is shown the 6 paired-stimulus input patterns presented during training (using the same conventions as in Fig. 2), while on the right is shown the synaptic weight matrix within the network after training. In the weight matrix plots, dark shading indicates a high weight value. In the plot of the synaptic weights, the output cells have been ordered to reveal the underlying weight structure which developed during training. The synaptic weights show that the output cells have typically learned to respond to pairs of stimuli, which correspond to the 6 original paired-stimulus input patterns presented during training. The bottom row shows results for the case $N = 10$ stimuli with $a = 0.2$. On the left are shown some of the 45 paired-stimulus training patterns, while on the right is shown the synaptic weight matrix after training. As above, the output cells have been ordered to reveal the underlying weight structure which developed during training. It can be seen that the weight matrix has a block-diagonal structure, which clearly shows that the output cells have each learned to respond to one particular stimulus.

Table 1
A summary of how the output cells of the 1-layer competitive network learn to represent the input data when different numbers of independent stimuli N are used to generate multi-stimulus training patterns

Number of stimuli N	Number of cells which respond to 1 stimulus	Number of cells which respond to 2 stimuli
3	0.2	18.0
4	0.0	36.0
5	0.0	60.0
6	12.3	0.0
7	18.8	0.0
8	27.5	0.2
9	35.7	0.3
10	44.8	1.5

For each value of N, average results were obtained over 6 simulation runs. For these experiments, the sparseness of the output firing rates a was always set to 0.05. It can be seen that when there are a small number of stimuli, i.e. for $N = 3$ to 5, the output cells typically learn to respond to pairs of stimuli, corresponding to the multi-stimulus input patterns which the network was exposed to during training. However, when the number of stimuli N is increased to 6 or more, the output cells typically learn to respond to only a single one of the stimuli. Thus, for large numbers of stimuli N, the output cells of the competitive network learn to represent the N individual independent stimuli rather than the $N(N-1)/2$ multi-stimulus training patterns, which the network was exposed to during training.

Table 2
A summary of how the output cells of the 1-layer competitive network learn to represent the input data when the level of mutual inhibition within the output layer is varied in order to control the sparseness of the output firing rates a

Sparseness of output firing a	Number of cells which respond to 1 stimulus	Number of cells which respond to 2 stimuli
0.01	11.2	0.0
0.02	14.7	0.0
0.05	45.5	1.0
0.1	69.3	9.3
0.2	100.0	0.0
0.5	66.3	0.3

For these experiments, the number of independent stimuli N was always set to 10. It can be seen that, with a relatively large number of $N = 10$ stimuli, the output cells mostly learned to respond to only a single one of the stimuli across a wide range of values of the sparseness of the output firing rates a. However, the number of output cells which responded to a single stimulus was maximized when the output sparseness a was set to 0.2, where it can be seen that all 100 output cells are responsive to only a single stimulus.

Examples of the firing rates of the output neurons in these simulations after training are shown in Fig. 3 (obtained for the case, $N = 10$ stimuli and the output firing rate sparseness set to 0.2). In each plot, the cell responds to only one of the 10 stimuli.

The behavior of the competitive network can be further understood from the synaptic weight matrices shown in Fig. 4. The top row shows results for the case, $N = 4$ stimuli with $a = 0.14$. The synaptic weights show that the output cells have typically learned to respond to pairs of stimuli, which correspond to the 6 original paired-stimulus input patterns presented during training. The bottom row shows results for the case, $N = 10$ stimuli with $a = 0.2$. It can be seen that the weight matrix has a block-diagonal structure, which clearly shows that the output cells have each learned to respond to only one of the independent stimuli. Since the blocks on the diagonal are almost square, the weight matrix also confirms that the output cells are distributed fairly evenly among the 10 stimuli, with disjoint subsets of approximately 10 output cells having learned to respond to each of the 10 independent stimuli.

The simulations described above were performed with 10,000 training epochs to ensure convergence during learning. However, similar effects can still be observed after far fewer training epochs. We repeated an experiment with $N = 10$ independent stimuli and the sparseness of the output firing rates set to 0.2. However, this time we increased the learning rate to $k = 0.01$ and used only 100 training epochs. This led to identical results to those shown in row 5 of Table 2. Thus, 100 training epochs were enough to ensure that all of the output cells learned to respond to a single

stimulus. Further simulations showed that 5, or even as few as 1 training epoch, produced the same change of behavior with low N producing paired-stimulus representations, and $N \geq 6$ producing single-stimulus representations. Further simulations showed that the effects described were obtained with random order of the training patterns as well as a fixed order of the training patterns during each training epoch.

The reason why increasing the number of independent stimuli causes the competitive network to switch from learning to represent the paired-stimulus training patterns to representing the independent stimuli may be understood by examining how often individual input neurons are co-active during training. The number of times that the neurons within a stimulus are simultaneously active during one training epoch in which all stimulus pairs are presented is $N - 1$. In contrast, the neurons from different stimuli are co-active on only one occasion in a training epoch. The ratio of the within-stimulus to between-stimulus co-activations of input neurons is thus $(N - 1) : 1$ (where N is the number of stimuli or objects). Given that competitive networks effectively build categories based on correlations between sets of input neurons to the network, as N increases, a stage is reached where instead of representing the pairs of stimuli actually presented during training, the network instead forms representations of the individual independent stimuli. From Table 1 it can be seen that the switch from representing the paired-stimulus training patterns to representing the independent stimuli occurred at $N = 6$ stimuli.

The mechanism of this learning is clear in relation to the associations or correlations that are present between neurons that are part of the same single stimulus; and between neurons that are parts of different stimuli. With the stimulus set used, in the case of $N = 10$ and one whole training epoch with the pair-stimuli used during training, the afferent neurons for any one stimulus will have been co-active on 9 training trials, whereas the same afferent neurons for the same stimulus will have been co-active on only 1 trial with any of the other neurons relating to other stimuli. Thus for an output neuron that is activated by the first single stimulus, there will have been 9 occasions on which the active synapses are strengthened by associative learning, and only 1 trial on which any other set of synapses active for a different stimulus has been strengthened by associative learning. Thus in principle with a simple Hebbian associative synaptic modification rule (shown in Eq. (3)) the within single-stimulus synapses will be 9 times stronger than the synapses from the other stimuli. It is this that results in the output neurons building much stronger representations of each single stimulus than of the stimulus pairs presented during training, with $N = 10$. Because with large N the synapses become so much stronger onto a given output neuron by associativity than the synapses from the other single patterns presented during learning, the output of the system becomes dominated by the connections between the single-stimulus co-active afferents, and the system represents the single-stimulus patterns, even though during training they have been presented on all trials with other single-stimulus patterns. We note that with the weight normalization used (Eq. (4)), the smallest synaptic weights will approach zero.

4. Learning *transform-invariant* representations of multiple stimuli in a 1-layer competitive network

It is important for the visual system to be able to produce transform-invariant representations of objects, so that the same output representation can be activated by for example the different locations or views of an object (Riesenhuber & Poggio, 1999; Rolls & Deco, 2002). We now hypothesize that invariance learning could be combined with the mechanism for learning about individual objects described in Sections 2 and 3. We test this hypothesis in this Section. If the hypothesis is supported, this proposal provides a powerful mechanism for learning invariant representations of objects even in cluttered scenes composed of multiple objects.

Invariance learning can be implemented using the temporal continuity inherent in an object as it is transforming, or in the spatial continuity that is present from transform to transform as the object transforms continuously. In this paper, we take as an example the learning of invariances using spatial continuity that underlies Continuous Transformation learning (Stringer et al., 2006). First we describe Continuous Transformation learning (Section 4.1). Then we test whether it can be successfully combined with the mechanism for learning about individual objects described in Sections 2 and 3.

4.1. Continuous transformation (CT) learning

When competitive networks self-organize with spatially continuously transforming input stimuli, individual neurons can learn to respond to the different transforms of each stimulus using a purely associative (Hebbian) synaptic modification rule (Stringer et al., 2006). Continuous Transformation (CT) learning has been proposed as a way in which transform-invariant representations may develop in the ventral visual system (Stringer et al., 2006). The learning mechanism is quite general, and may operate in brain areas with the architecture of a competitive network that are trained with continuous patterns.

The Continuous Transformation learning process is illustrated in Fig. 5, and operates as follows. A stimulus is initially situated at a first location in the input layer, and a small number of input cells are activated. Activity is then propagated through feedforward synaptic connections, from the input cells to the layer of output cells. Next, a small winning set of neurons in the output layer will modify (through associative learning) their afferent connections from the input layer to respond well to the input cells which represent the stimulus in its first location or transform. However, there is spatial continuity in the way the stimulus transforms because the stimulus moves continuously through the input space, one neuron at a time. Because successive stimulus transforms overlap, many of the same input cells are still active when the stimulus is at a nearby location. These active input neurons with their strengthened synapses ensure that the same output neurons will be activated because some of the active afferents are the same as when the stimulus was in the first position. The key point is that if these afferent connections have been strengthened sufficiently while the stimulus is in the first location, then these connections will be able to continue to activate the same neurons in the output layer when the stimulus is situated in overlapping nearby locations. Then any newly active synapses have conjunctive pre- and post-synaptic activity because of the move of location of the stimulus, and these synapses are increased in strength by Hebbian associativity. As can be seen in Fig. 5, the process can be continued for subsequent shifts, provided that a sufficient proportion of input cells stay active between individual shifts of location. This process can lead individual output cells to learn to respond to a particular stimulus over all locations. A fuller description of Continuous Transformation learning, and simulation results in the context of invariant object recognition, is provided by Stringer et al. (2006) and Perry et al. (2006).

4.2. Learning transform-invariant representations of multiple stimuli

The hypothesis we explore next is that a 1-layer competitive network can combine the Continuous Transformation learning mechanism for learning transform-invariant representations (Stringer et al., 2006), with the learning dynamics described above in Sections 2 and 3 for developing separate representations

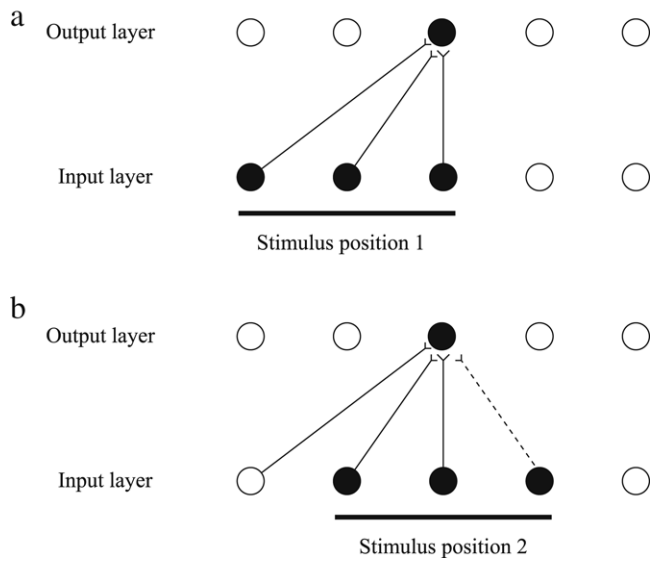

Fig. 5. An illustration of how Continuous Transformation learning would function in a network with a single layer of forward synaptic connections between an input layer of neurons and an output layer. Initially the forward synaptic weights are set to random values. The top part (a) shows the initial presentation of a stimulus to the network in position 1. Activation from the (shaded) active input cells is transmitted through the initially random forward connections to stimulate the cells in the output layer. The shaded cell in the output layer wins the competition in that layer. The representation is schematic, and many cells might be active after the competition. The weights from the active input cells to the active output neuron are then strengthened using an associative learning rule (an example of which is shown in Eqs. (3) and (4)). The bottom part (b) shows what happens after the stimulus is shifted by a small amount to a new partially overlapping position 2. As some of the active input cells are the same as those that were active when the stimulus was presented in position 1, the same output cell is driven by these previously strengthened afferents to win the competition again. The rightmost shaded input cell activated by the stimulus in position 2, which was inactive when the stimulus was in position 1, now has its connection to the active output cell strengthened (denoted by the dashed line). Thus the same neuron in the output layer has learned to respond to the two input patterns that have similar vector elements in common. As can be seen, the process can be continued for subsequent shifts, provided that a sufficient proportion of input cells stay active between individual shifts. During the learning, the synaptic weight vectors are normalized as is standard in competitive networks, so that no single neuron comes to dominate the competition, and different neurons can encode different stimuli.

of independent stimuli even when multiple stimuli are shown simultaneously. That is, we show that when a 1-layer competitive network is exposed to sequences of input patterns containing multiple transforming stimuli during training, the network is able to develop separate representations of the different stimuli, as well as simultaneously developing transform-invariant representations of those stimuli.

We consider in this section a simple example of invariance learning with multiple objects present on every trial. Each object can occur in a number of different transforms, with one transform of each object presented on each trial. The goal of the self-organizing (unsupervised) network is to map all transforms of each object to a category of output specific for that object. This thus provides a simple model of invariance learning. This has to be accomplished even when multiple (in this case two) objects are presented on every trial. Part of the difficulty of the problem is that the inputs that represent the most distant transforms of a given object do not overlap in the input space. We simulated this in the simplest model that would allow these processes to be investigated. This meant using non-overlapping representations of each object in a 1-layer network, though we have shown elsewhere that whether the representations overlap or not is not crucial to the type of learning with multiple objects being investigated here (Stringer, Rolls, & Tromans, 2007). We emphasize that two sets of afferents, one set for each object, are active on every learning trial; and the problem to be solved is how to map these simultaneously presented inputs to different output categories.

4.3. Model and stimuli

The neural network architecture used in this Section remains similar to that described in Section 3 above and shown in Fig. 1. However, the input patterns are now altered to allow the stimuli to be presented in different transforms, as described next.

In the 1-layer competitive network simulations with transform-invariant representations, there are $N = 10$ independent stimuli, each of which can occur in 11 transforms. (In Section 3 it was shown that for $N = 10$ stimuli the network was able to develop separate representations of the individual stimuli.) Fig. 6 shows schematically the input representations of the 11 transforms of each of the 10 independent stimuli.

During training, the network is presented with every possible pair of stimuli. The network is presented with training sequences of 11 consecutive transforms for each stimulus pair, with the two stimuli transforming together in lockstep. That is, for each pair of stimuli, the network is presented with a sequence of 11 input patterns which contain the 11 consecutive transforms of each stimulus. Examples of these stimulus-pair training sequences are shown on the left of Fig. 8. The presentation of the 11 transforms of all possible $N(N-1)/2$ stimulus pairs corresponded to one training epoch. There were 10,000 training epochs to ensure convergence of the synaptic weights. After training, the network was tested with all 11 transforms of each of the $N = 10$ stimuli.

4.4. Simulation results

The results described next confirm that the network can successfully combine the Continuous Transformation learning mechanism for learning transform-invariant representations with the learning dynamics described above in Sections 2 and 3 for developing separate representations of independent stimuli. After training the network, all of the output cells were found to be responsive to only one of the independent stimuli. Furthermore, each output cell responded to all of the 11 transforms of its favoured stimulus. It was also found that approximately equal numbers of output cells responded to each of the stimuli (as is also evident from the weight matrix shown in Fig. 8).

Fig. 7 shows the firing rate responses of two typical output cells, after the network has been trained on pairs of $N = 10$ stimuli, each of which was shifted continuously through 11 transforms (i.e. locations) during learning. The sparseness of the output firing rates was set to 0.2 during training and testing. For this test case, each of the 100 output cells learned to respond to all transforms of one particular stimulus, but did not respond to any of the transforms of the other stimuli. In this way, each of the output cells developed a transform-invariant representation of one of the stimuli. Furthermore, disjoint subsets of approximately 10 output cells learned to respond in a transform-invariant way to each of the independent stimuli. Thus, the $N = 10$ input stimuli were all equally well represented by the output layer of the network. The upper two rows show the response of output cell 9 to each of the $N = 10$ stimuli in each of their 11 transforms. It can be seen that output cell 9 responds to stimulus 3 over all 11 transforms, but does not respond to any of the other stimuli in any location. The lower two rows show the response of output cell 8 to each of the $N = 10$ stimuli in each of their 11 transforms. It can be seen that output cell 8 responds to stimulus 7 over all 11 transforms, but does not respond to any of the other stimuli in any location.

Fig. 8 shows the synaptic weights after the training. On the left are shown some of the paired-stimulus training patterns. On the

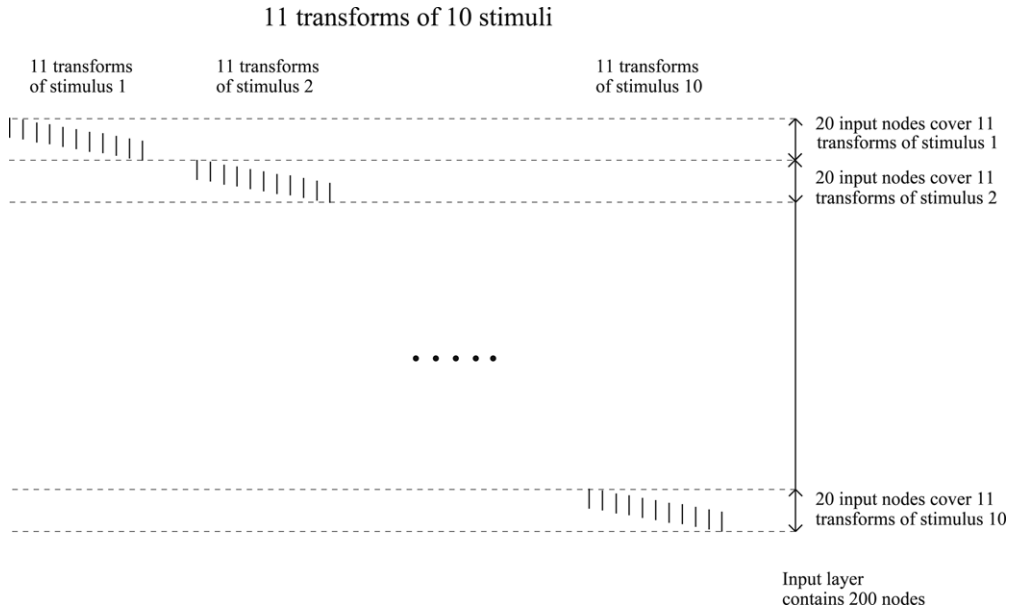

Fig. 6. In the 1-layer competitive network simulations with transform-invariant representations, there are $N = 10$ independent stimuli, each of which can occur in 11 transforms. Here we show the input representations of the 11 transforms of each of the 10 independent stimuli (using the same conventions as in Fig. 2). (The diagram explicitly shows the 11 transforms of stimuli 1, 2 and 10. The intervening stimuli 3 to 9 transform similarly.) The 10 stimulus patterns are each represented by the activation of a contiguous block of 10 cells. Each stimulus undergoes transformation by continuously shifting the location of the stimulus by one input neuron at a time. During training, each stimulus is shifted through 11 overlapping locations. Given that each stimulus is 10 neurons long, this means that the 11 transforms of each stimulus cover 20 input neurons. This ensures that the first and last transforms of each 10-neuron stimulus do not overlap. In order to ensure that none of the input representations of the 10 different stimuli overlap, the 11 transforms of each stimulus cover a unique disjoint block of 20 input neurons. The input layer is thus set to contain 200 neurons in order to be able to represent all of the transforms of all of the stimuli.

right is shown the synaptic weight matrix after training. For the weight matrix plot, the output cells have been ordered to reveal the underlying weight structure which developed during training. It can be seen that the weight matrix has a block-diagonal structure, which clearly shows that each of the output cells has learned to respond to all 11 transforms of one particular stimulus, but has not learned to respond to any of the other stimuli. Since the blocks on the diagonal are almost square, the weight matrix also confirms that the output cells are distributed fairly evenly among the 10 stimuli, with disjoint subsets of approximately 10 output cells having learned to respond in a transform-invariant way to each of the 10 independent stimuli.

Although the above simulation used 10,000 training epochs, similar results were also obtained with only 5 training epochs as long as the learning rate was increased to $k = 0.01$. Over six simulation runs with 5 training epochs, an average of 61.0 output cells (with standard error 2.4) still learned to respond to all transforms of one particular stimulus, but did not respond to any of the transforms of the other stimuli.

During invariance learning, the presentation of one object in the real world might be interrupted by another stimulus, and it is of interest to discover whether transform-invariant learning can operate when objects are interleaved during training. This is a property of continuous transformation learning (Stringer et al., 2006), but has not been investigated before when there is more than one object present during training. In further simulations, on any one training epoch the network was firstly trained with images containing the first transforms of every possible pair of objects, then with images containing the second transforms of every possible pair of objects. A learning similar to that already described was found, with the synaptic weight matrix almost identical to that shown in Fig. 8. Six simulation runs in total were performed with different initial random synaptic weights. For all six of the simulations, each of the 100 output cells learned to respond to all transforms of one particular object, but did not respond to any of the transforms of the other objects. In this way, each of the output cells developed a transform-invariant representation of one of the objects. Furthermore, in all simulations, disjoint subsets of approximately 10 output cells learned to respond in a transform-invariant way to each of the independent objects. Thus, the $N = 10$ input stimuli were all equally well represented by the output layer of the network. Although these simulations used 1,000 training epochs, similar results were also obtained with only 5 training epochs. Over six simulation runs with 5 training epochs, an average of 83.0 output cells out of 100 (with standard error 1.6) learned to respond to all transforms of one particular object, but did not respond to any of the transforms of the other objects. Also, in three of the simulations there were no output cells which responded to more than one object, while in the other three simulations there was only one cell in each simulation which responded to more than one object over all transforms. This experiment establishes that even when different objects are interleaved during training, the network can learn perfect invariant representations of objects when trained with an associative (Hebb) rule with pairs of objects present on every training trial in the scene.

During invariance learning, some of the transforms of an individual object might not overlap at all in terms of the neurons that are activated by each transform of the same object. Therefore, an important question is whether continuous transformation learning with a purely associative learning rule (which cannot take advantage of the temporal proximity of non-overlapping transforms of the same object) can still operate effectively when the order of stimulus transforms is randomized. In this case, spatially similar transforms of an individual object do not necessarily occur close together in the training sequence. In further simulations we showed that even when the order of stimulus transforms is randomized so that spatially similar transforms of an individual object do not occur close together, the system can nevertheless use the spatial continuity present across the whole set of transforms of an object to learn an invariant representation of that object. This can occur even when pairs of objects are presented in a scene on each trial. The performance of the network was

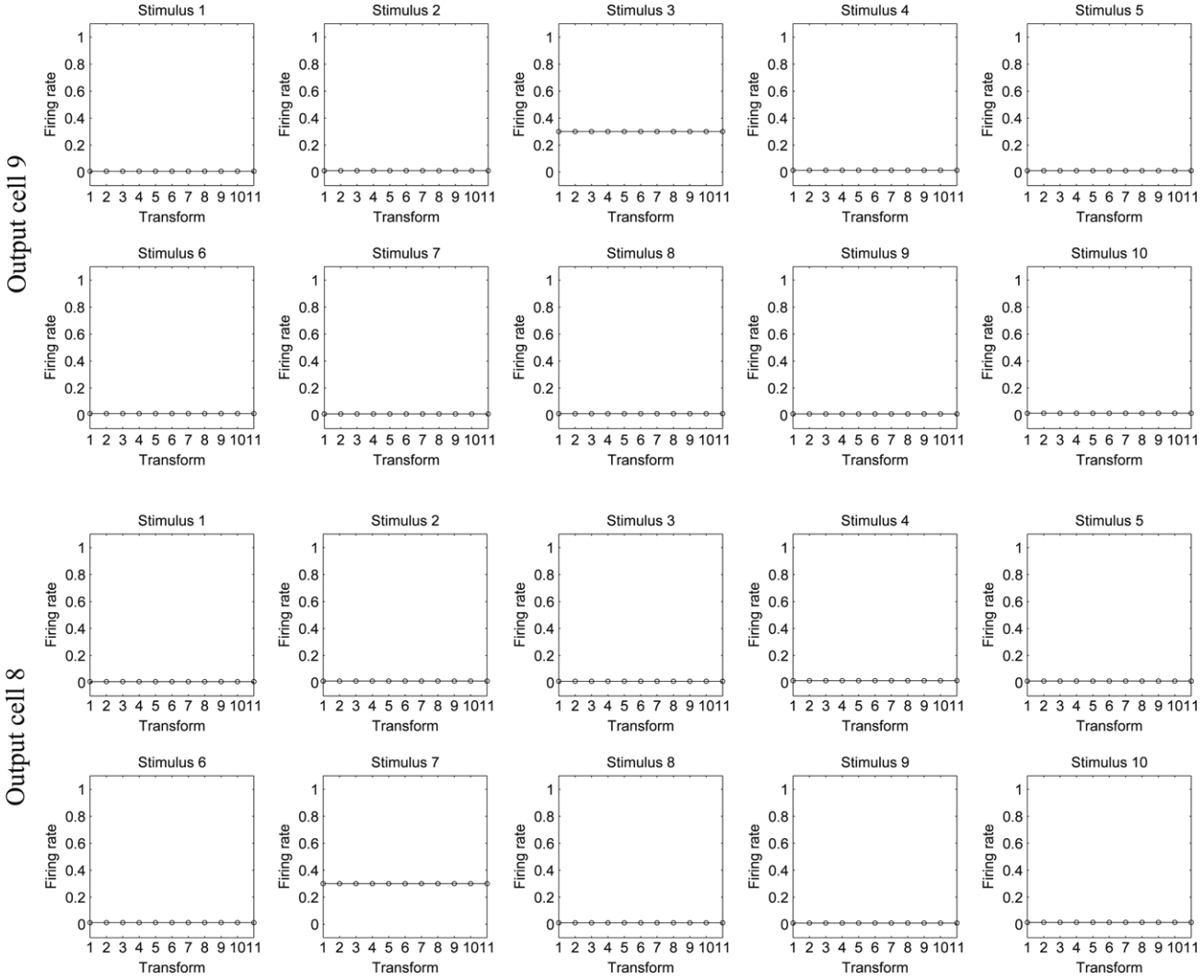

Fig. 7. The firing rate responses of two typical output cells, after the network has been trained on pairs of $N = 10$ stimuli, each of which was shifted continuously through 11 transforms (i.e. locations) during learning (see the text).

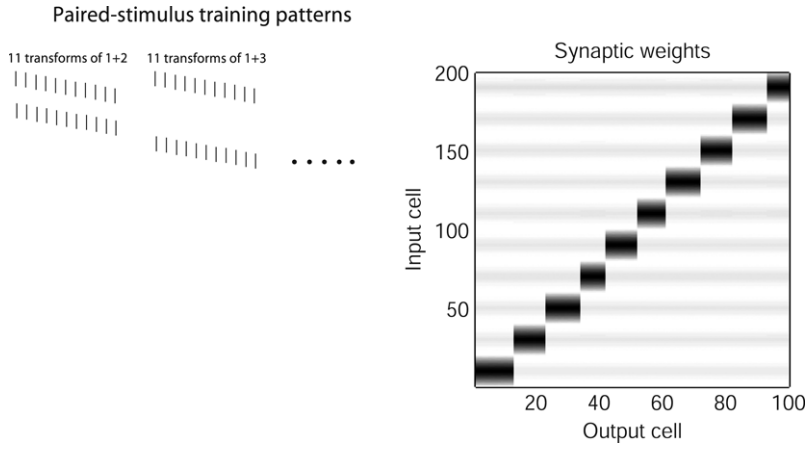

Fig. 8. The synaptic weights after the network has been trained on pairs of $N = 10$ stimuli, each of which was shifted continuously through 11 transforms (i.e. locations) during learning. The sparseness of the output firing rates was set to 0.2 during training and testing. On the left are shown some of the paired-stimulus training patterns (using the same conventions as in Fig. 2). As an example, we have shown 11 transforms of stimulus pair $1 + 2$ and 11 transforms of stimulus pair $1 + 3$. On the right is shown the synaptic weight matrix after training, where dark shading indicates a high weight value. For the weight matrix plot, the output cells have been ordered to reveal the underlying weight structure which developed during training.

similar to that described above with interleaved object transforms during training, with the synaptic weight matrix almost identical to that shown in Fig. 8. In each of six simulations, it was found that an average of 96.8 output cells (with standard error 0.83) still learned to respond to all transforms of one particular object, but did not respond to any of the transforms of the other stimuli.

It was also found that approximately equal numbers of output cells responded to each of the objects. Similar results were also obtained with only 5 training epochs. Over six simulation runs with 5 training epochs, an average of 61.8 output cells out of 100 (with standard error 3.9) learned to respond to all transforms of one particular object, but did not respond to any of the transforms of the other objects. No output cells were found in any of the six simulations which responded to more than one object. This experiment thus shows that in the situation described in this paper of multiple stimuli always present in a scene during training, the competitive network can nevertheless learn invariant representations of each object, even when the transforms are presented in random order, and the extreme transforms do not overlap.

The results in this Section thus show that invariant representations of a stimulus can be learned even when there are multiple stimuli simultaneously present during training. The network learns separate representations of the different stimuli because the set of inputs that characterize each object are more frequently present with each other than they are with the active inputs that represent different stimuli, even when different transforms of the objects are occurring as part of the training.

5. Training a multi-layer feedforward network (VisNet) with multiple 3D rotating stimuli

We now show how the learning mechanisms described above can operate in a more biologically realistic model of transform (e.g. view) invariant object recognition in the ventral visual processing stream, VisNet (Wallis & Rolls, 1997). To do this, we trained on a view invariance problem, in which each stimulus was a 3D object shown in 90 views, and the aim was to test whether the network could form view-invariant representations of individual objects. What was different from any previous training attempted with VisNet was that whenever one object was presented in its sequence of rotations, another object was also presented simultaneously, and also rotating in the same way. This enabled us to test whether VisNet could form invariant representations of each object. The nature of the object pairing was of the same form described above, that is each object was paired during training with every other object.

The model architecture (VisNet) (Wallis & Rolls, 1997) is based on the following: (i) A series of hierarchical competitive networks with local graded inhibition. (ii) Convergent connections to each neuron from a topologically corresponding region of the preceding layer, leading to an increase in the receptive field size of neurons through the visual processing areas. (iii) Synaptic plasticity based on a Hebb-like learning rule to allow CT invariance learning. In previous work, model simulations which incorporated these hypotheses were shown to be capable of producing stimulus-selective but translation and view-invariant representations of the trained stimuli (Stringer et al., 2006). However, in these earlier simulations, the stimuli were always presented one at a time during learning. We now show how the VisNet architecture is able to learn view-invariant representations of individual stimuli when multiple stimuli are presented together during training.

The model consists of a hierarchical series of four layers of competitive networks, corresponding to V2, V4, the posterior inferior temporal cortex, and the anterior inferior temporal cortex, as shown in Fig. 9. The forward connections to individual cells are derived from a topologically corresponding region of the preceding layer, using a Gaussian distribution of connection probabilities. These distributions are defined by a radius which will contain approximately 67% of the connections from the preceding layer. The values used are given in Table 3.

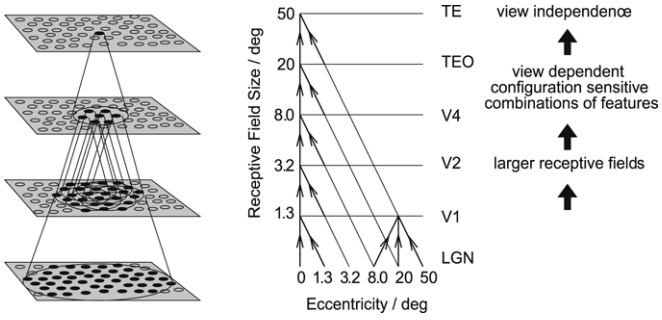

Fig. 9. Left: Stylized image of the 4-layer network. Convergence through the network is designed to provide 4th layer neurons with information from across the entire input retina. Right: Convergence in the visual system V1: visual cortex area V1; TEO: posterior inferior temporal cortex; TE: inferior temporal cortex (IT).

Table 3
Network dimensions showing the number of connections per neuron and the radius in the preceding layer from which 67% are received

	Dimensions	Number of connections	Radius
Layer 4	32 × 32	100	12
Layer 3	32 × 32	100	9
Layer 2	32 × 32	100	6
Layer 1	32 × 32	272	6
Retina	128 × 128 × 32	–	–

Table 4
Layer 1 connectivity

Frequency	0.5	0.25	0.125	0.0625
Number of connections	201	50	13	8

The numbers of connections from each spatial frequency set of filters are shown. The spatial frequency is in cycles per pixel.

Before stimuli are presented to the network's input layer they are pre-processed by a set of input filters which accord with the general tuning profiles of simple cells in V1. The input filters used are computed by weighting the difference of two Gaussians by a third orthogonal Gaussian according to the following:

$$\Gamma_{xy}(\rho,\theta,f) = \rho\left[e^{-(\frac{x\cos\theta+y\sin\theta}{\sqrt{2}/f})^2} - \frac{1}{1.6}e^{-(\frac{x\cos\theta+y\sin\theta}{1.6\sqrt{2}/f})^2}\right]e^{-(\frac{x\sin\theta-y\cos\theta}{3\sqrt{2}/f})^2}, \quad (5)$$

where f is the filter spatial frequency, θ is the filter orientation, and ρ is the sign of the filter, i.e. ± 1. Individual filters are tuned to spatial frequency (0.0625 to 0.5 cycles/pixel); orientation (0° to 135° in steps of 45°); and sign (± 1). The number of layer 1 connections to each spatial frequency filter group is given in Table 4.

The activation h_i of each neuron i in the network is set equal to a linear sum of the inputs y_j from afferent neurons j weighted by the synaptic weights w_{ij}. That is,

$$h_i = \sum_j w_{ij} y_j, \quad (6)$$

where y_j is the firing rate of neuron j, and w_{ij} is the strength of the synapse from neuron j to neuron i.

Within each layer competition is graded rather than winner-takes-all, and is implemented in two stages. First, to implement lateral inhibition the activation h of neurons within a layer is converted to firing rates r using a linear activation function followed by convolution with a spatial filter, I, where δ controls the contrast and σ controls the width, and a and b index the distance away from the centre of the filter in orthogonal directions

$$I_{a,b} = \begin{cases} -\delta e^{-\frac{a^2+b^2}{\sigma^2}} & \text{if } a \neq 0 \quad \text{or } b \neq 0, \\ 1 - \sum_{\substack{a\neq 0 \\ b\neq 0}} I_{a,b} & \text{if } a = 0 \quad \text{and} \quad b = 0. \end{cases} \quad (7)$$

Table 5
Lateral inhibition parameters

Layer	1	2	3	4
Radius, σ	1.38	2.7	4.0	6.0
Contrast, δ	1.5	1.5	1.6	1.4

Table 6
Sigmoid parameters

Layer	1	2	3	4
Percentile	99.2	98	88	91
Slope β	190	40	75	26

The lateral inhibition parameters are given in Table 5.

The modelling of lateral inhibition incorporates also contrast enhancement (related to nonlinearity in the neurons) which is applied by means of a sigmoid activation function

$$y = f^{\text{sigmoid}}(r) = \frac{1}{1 + e^{-2\beta(r-\alpha)}}, \tag{8}$$

where r is the firing rate after the lateral inhibition filter defined above, y is the firing rate after contrast enhancement, and α and β are the sigmoid threshold and slope respectively. The parameters α and β are constant within each layer, although α is adjusted to control the sparseness of the firing rates. For example, to set the sparseness to, say, 5%, the threshold is set to the value of the 95th percentile point of the activations within the layer. The parameters for the sigmoid activation function are shown in Table 6.

At each timestep, the activity due to the stimulus on the retina is propagated in a feedforward fashion through the network, stimulating patterns of activity in the later layers. Once the activity patterns have been computed in the various layers including competitive lateral inhibition as described above, the synaptic weights of the forward connections between the layers are updated by an associative learning rule which enhances the synaptic weight between two neurons when they are co-firing. There are a variety of associative rules that could be used. In the simulations described in this paper with VisNet we used CT learning (see Section 4.1) and therefore the Hebb associative learning rule

$$\delta w_{ij} = \alpha y_i x_j, \tag{9}$$

where δw_{ij} is the increment in the synaptic weight w_{ij}, y_i is the firing rate of the post-synaptic neuron i, x_j is the firing rate of the pre-synaptic neuron j, and α is the learning rate. To bound the growth of each neuron's synaptic weight vector, $\mathbf{w}_i$ for the ith neuron, its length is normalized at the end of each timestep during training as in usual competitive learning (Hertz et al., 1991).

5.1. Stimuli

The stimuli used to train the network were images of $N = 9$ continuously rotating 3D objects. OpenGL was used to build a 3D representation of the objects, and then to project different views onto a 2D image. There was ambient lighting with a diffuse light source added to allow different surfaces to be shown with different intensities. The nine stimuli are shown in Fig. 10. Each of the stimuli was presented in the locations shown. In the experiments described here, nine stimuli were used because this was found to be sufficient in Section 3 above to allow a competitive network to develop representations of individual stimuli when the network was trained on stimulus pairs.

Given nine stimuli, there are 36 possible pairs of stimuli to train the network on. Therefore, during training, the network was presented with 36 image sequences, where each image sequence corresponded to one of the 36 possible pairs of stimuli.

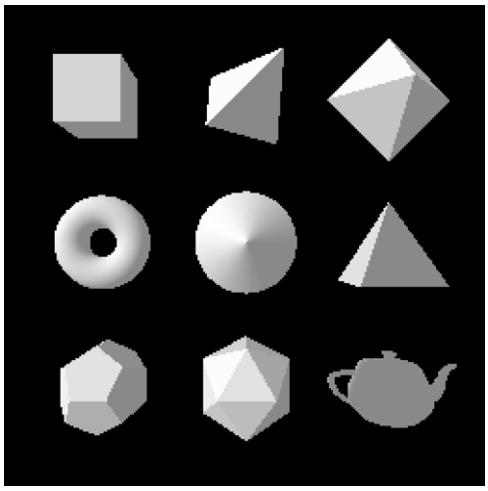

Fig. 10. The nine stimuli used to train the VisNet network. The stimuli were 3D objects, each shown from 90 different views in a test of view-invariant recognition. The effect of the ambient lighting and a single diffuse light source to allow different surfaces to be shown with different intensities is illustrated. In rows from left to right the stimuli were a cube, tetrahedron, octahedron, torus, cone, pyramid, dodecahedron, icosahedron and a teapot. During training, the network was presented with every possible pair of stimuli simultaneously rotating round their vertical axis over a range of 90°, with a step size of 1° between successive views. After training, the network was tested with image sequences of the nine individual stimuli rotating through 90°.

For each image sequence during training, the two stimuli were simultaneously rotated about their vertical axis over a range of 90°, with a step size of 1° between successive views. The small step size is necessary to allow CT learning to operate. Fig. 11 shows three example frames selected from the 90-frame training image sequence of the cube and the pyramid rotating through 90 degrees in 1° steps. The selected frames shown in Fig. 11 are for 30°, 60° and 90°. In total, there are 36 such image sequences used during training, with each image sequence corresponding to one of the 36 possible pairs of stimuli rotating through 90° in 1° steps.

After training the network, we tested it by recording the firing rates of the neurons in the 4th (output) layer of the network as the network was presented with image sequences of the nine individual stimuli rotating through 90°.

5.2. Training and test procedure

To train the network each pair of stimuli is presented to the network in a sequence of different transforms (i.e. views). At each presentation the activation of individual neurons is calculated, then their firing rates are calculated, and then the synaptic weights are updated. The presentation of all the stimulus pairs across all transforms constitutes 1 epoch of training. In this manner the network is trained one layer at a time starting with layer 1 and finishing with layer 4. In the investigation described here, the numbers of training epochs for layers 1–4 were 50, 100, 100 and 75 respectively. The learning rate α in Eq. (9) was 0.0018 for layer 1, and was 0.005 for layers 2–4.

An information theoretic measure of performance was used to assess the ability of the output layer of the network to develop neurons that are able to respond with view invariance to individual stimuli or objects (see Rolls and Milward (2000)). This single cell information measure was applied to individual cells in layer 4 and measures how much information is available from the response of a single cell about which stimulus was shown independently of the view. For each cell the single cell information measure used was the maximum amount of information a cell conveyed about any one stimulus. This is computed using the following formula with

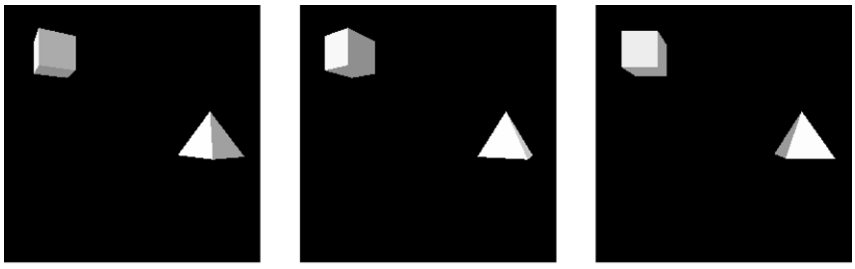

Fig. 11. Three example frames selected from the 90-frame training image sequence of the cube and the pyramid rotating through 90° in 1° steps. The selected frames shown are for 30°, 60° and 90°. In total, there are 36 such image sequences used during training, with each image sequence corresponding to one of the 36 possible pairs of stimuli rotating through 90° in 1° steps.

details given by Rolls, Treves, Tovee, and Panzeri (1997) and Rolls and Milward (2000). The stimulus-specific information or surprise $I(s, R)$ is the amount of information the set of responses R has about a specific stimulus s, and is given by

$$I(s, R) = \sum_{r \in R} P(r|s) \log_2 \frac{P(r|s)}{P(r)}, \tag{10}$$

where r is an individual response from the set of responses R. The maximum single cell information measure is

$$\text{Maximum single cell information} = \log_2(\text{Number of stimuli}), \tag{11}$$

where in this case the number of stimuli is 9. This gives a maximum single cell information measure of 3.17 bits, and is achieved when all the responses to the different views of a stimulus are similar, that is when the representation is invariant, and when the neuron does respond to any other stimulus.

5.3. VisNet simulation results

After the network had been trained on the 36 pairs of stimuli each rotating over a range of 90°, many cells in the 4th (output) layer were found to have learned to respond to one of the nine stimuli over all views, but did not respond to any of the other stimuli from any view, as described next. In particular, neurons that responded to a particular stimulus after training had no response to any of the other stimuli, even though the other stimuli had been presented simultaneously with the particular stimulus during training.

Fig. 12 shows the firing rate responses of cell (10, 13) in the 4th (output) layer to all views of the nine individual stimuli before and after training. The top nine plots show the responses of cell (10, 13) to each of the nine stimuli as they are rotated through 90° in 1° steps before training. It can be seen that before training the cell responds randomly to a number of the stimuli over small regions of the view space of different stimuli. The bottom nine plots show the responses of cell (10, 13) to each of the nine stimuli after training with the 36 possible pairs of stimuli rotating through 90°. It is evident that after training, the cell has learned to respond to the cube over all views, but does not respond to any of the other stimuli in any views. The cell has thus learned to respond view invariantly to the cube, even though during training the cube was presented on different trials with other members of the training set of stimuli.

Complementary learning by a different cell (17, 12) is illustrated in Fig. 13. It is evident that after training, the cell has learned to respond to the pyramid over all views, but does not respond to any of the other stimuli in any views. The cell has thus learned to respond view invariantly to the pyramid. Thus different cells learn to respond to different stimuli.

These two cells thus learned separate invariant representations of the cube and pyramid, which had been presented together within single training images (of object pairs) as shown in Fig. 11.

Fig. 14 shows the information results obtained when VisNet was tested with the nine individual stimuli rotating through 90° in 1° steps. Results are presented after training the network with image sequences of all 36 possible stimulus pairs (unbroken line), and with a random untrained network (dashed line). The single cell information measures for all top (fourth) layer neurons ranked in order of their invariance to the stimuli are shown. It can be seen that training the network on the stimulus pairs has led to many top layer neurons attaining the maximal level of single cell information of 3.17 bits. This corresponds to a neuron responding to only one of the 9 stimuli, and indeed producing a similar response to every view of that stimulus. If a neuron had responded to only some of the views of that stimulus, or had responded to any of the views of another stimulus, the single cell information would have been less than 3.17 bits. Thus the information theory analysis shown in Fig. 14 shows that many of the top layer neurons have learned to respond to all views of one stimulus, and to no views of any other stimulus, even though each stimulus was being trained in the presence of one other stimulus in the scene (permuted from the other stimuli in the set). Moreover, the multiple cell information result reached 3.17 bits, showing that the neurons in the output layer had learned perfect invariant discrimination between the stimuli. The correct learning to objects in this hierarchical network model of the ventral visual stream occurs because during training of any one object, the features of that object always co-occur, and this leads the competitive networks in the hierarchy to form representations of these co-active inputs. The neurons do not learn to respond to the other stimuli being presented simultaneously, because there are 8 other stimuli, each of which is only sometimes presented with the first object, so that the features of these other objects are not presented very frequently with the features of the first object. These results show that training the network on the stimulus pairs has led to over half of the total number of output neurons each learning to respond to one of the nine stimuli over all possible 90 views.

6. Discussion

In natural vision an important problem is how the brain can build invariant representations of individual objects even when multiple objects are present in a scene. What processes enable the learning to proceed for individual objects rather than for the combinations of objects that may be present during training? In this paper we have described and analyzed an approach to this that relies on the statistics of natural environments. In particular, the features of a given object tend to co-occur with high probability with each other during training, whereas because during training this object may be seen with different objects