International Journal of Neural Systems, Vol. 2, Nos. 1 & 2 (1991) 17–33
© World Scientific Publishing Company

UNSUPERVISED LEARNING PROCEDURES FOR NEURAL NETWORKS

Suzanna Becker
Department of Computer Science, University of Toronto
Toronto, Ontario, M5S 1A4, Canada

Received 14 December 1990
Revised 26 February 1991

Supervised learning procedures for neural networks have recently met with considerable success in learning difficult mappings. However, their range of applicability is limited by their poor scaling behavior, lack of biological plausibility, and restriction to problems for which an external teacher is available. A promising alternative is to develop unsupervised learning algorithms which can adaptively learn to encode the statistical regularities of the input patterns, without being told explicitly the correct response for each pattern. In this paper, we describe the major approaches that have been taken to model unsupervised learning, and give an in-depth review of several examples of each approach.

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

1. Introduction

A major goal of artificial intelligence research is to develop computational models which exhibit performance comparable to that of humans on the everyday information processing tasks we engage in as we interact with the world. The sort of tasks at which people are good, and (so far) at which machines are very poor, involve extracting meaning from complex patterns of stimuli such as the structure of objects in the visual world, words from acoustic signals, or sentence meaning from text. Human performance on these tasks is characterized by a robustness to noise and ambiguity, and a remarkable efficiency in situations which appear to require the simultaneous satisfaction of many constraints.

What are the critical features of the brain's algorithms and architecture which give rise to these abilities? A common assumption of the Connectionist modeling approach is that two necessary features of an artificially intelligent system are (1) a highly parallel architecture like the brain's and (2) the ability to learn from experience. Many connectionist models of learning have been proposed. This paper reviews recent work on unsupervised learning. We begin with a brief discussion of supervised learning algorithms, and how some of the problems with this type of learning may be overcome by using unsupervised learning procedures. We then give an in-depth review of various approaches that have been taken to develop unsupervised learning algorithms, and finally, we suggest future directions for work in this area.

2. Supervised versus Unsupervised Learning

The most widely used and successful supervised learning procedure for multilayer feedforward networks is the back-propagation algorithm.[1] Back-propagation learning has been successfully applied to some difficult problems, such as speech generation from text,[2] phoneme recognition,[3] and handwritten digit recognition.[4] Unfortunately, supervised learning algorithms have so far been limited by their poor scaling behavior: the learning becomes unacceptably slow as the size of the network or problem increases. This is particularly true of large nonlinear networks with many hidden layers. To understand this, consider that the effect of a weight in the first layer on the output of an m-layer network depends on its interactions with roughly $(fan\text{-}in)^m$ other weights, where *fan-in* is the average number of incoming links to units in the network. Hence, the parameter tuning problem in a large system with multiple stages of nonlinearities can take a prohibitively long time. Another limitation of current supervised learning procedures is with respect to biological plausibility; it is unlikely that the brain directly employs a back-propagation mechanism to train its "hidden units," nor is there any evidence suggesting that explicit target values are provided to individual neurons during learning.

One solution to these problems may be to make use of unsupervised learning procedures. In the unsupervised learning paradigm, rather than providing explicit examples of the function to be learned by the network, we provide a task-independent measure of the quality of the representation to be learned; we can then optimize the network parameters with respect to that measure. If we can apply the self-organizing process sequentially, one layer at a time, we can train deep networks in time linear in the number of layers. Once the network has become tuned to the statistical regularities of the input patterns, it is able to form internal representations which encode features of the input in a more explicit or simple form. We can now apply some more difficult (possibly supervised) learning problems, such as learning to navigate across rough terrain, or to interpret continuous speech sounds, to the transformed input. The hope is that this transformed representation of the sensory input will be easier to interpret, so that now we can quickly learn to associate the correct responses with the network's internal representation of the world.

There has been much recent progress in developing computational models of self-organizing processes in the brain. In the remaining sections of this paper, we review several different approaches to modeling unsupervised learning.

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

3. Hebbian Learning

The earliest proposal for an explicit rule of synaptic modification was made by Donald Hebb.[5] Hebb's now famous postulate about the mechanism of learning is as follows:

> *When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.*[5]

Now widely referred to simply as Hebbian learning, this rule has been used as the basic building block of associative memory in a wide variety of unsupervised learning models, as we shall see. The most direct computational expression for Hebb's rule causes the strength of a weight w_{ji} from unit i to unit j to increase in direct proportion to the product of presynaptic and postsynaptic activities:

$$w_{ji}(t) = w_{ji}(t-1) + \varepsilon y_i(t) y_j(t) .$$

One problem with this rule is that, assuming activities are always positive, the connection strengths can increase without bound. Many other forms of the Hebbian learning rule have been proposed which attempt to model various aspects of associative learning[6–14]; we describe two of these models in some detail.

Bienenstock *et al.*[7] proposed a variation of the simple Hebbian learning rule which results in a form of temporal selectivity. The basic idea is that in order for a single unit to compute some useful function it should respond selectively to some input patterns and not others, with respect to some particular environment. They proposed the following measure of selectivity, which depends on the ratio of the unit's mean response over all inputs to its maximal response:

$$\mathrm{Sel}(y_j) = 1 - \frac{\overline{y_j}}{\max(y_j)} .$$

The ideal unit, by this measure, gives a maximal response to one particular pattern and very low responses to the other patterns. To achieve this, the authors proposed a family of Hebbian learning rules which cause a weight w_{ij} to change in proportion to the product of the presynaptic activity y_i for that link and some function Φ of the postsynaptic activity y_j. Φ is chosen so that the sign of the weight change reverses when y_j is sufficiently large relative to the mean response $\overline{y_j}$. The general form of their learning rule is:

$$\dot{w}_{ji} = \Phi(y_j, \overline{y_j}) y_i - \varepsilon w_{ji}$$

where $\dot{w}$ denotes the rate of change of w over time. The rightmost term causes weights to decay exponentially toward zero in the absence of inputs. The function Φ must be chosen so that the unit's state converges to some stable equilibrium point of high selectivity. The authors show that Φ must therefore have the following properties:

$$\mathrm{sgn}[\Phi(y_j, \overline{y_j})] = \mathrm{sgn}\left[y_j - \left(\frac{\overline{y_j}}{c_o} \right)^p \overline{y_j} \right] \quad \text{if } y_j > 0$$

$$\Phi(0, \overline{y_j}) = 0 \quad \text{otherwise}$$

where p and c_o are positive constants, and units' activities are always positive. This says that when the unit's response is uniformly low ($\overline{y_j} \ll c_o$), the weight changes for almost all patterns should be positive. If

the mean response is very large ($\overline{y_j} \gg c_o$), on the other hand, all the weight changes should be negative. At some intermediary point ($\overline{y_j} \approx c_o$), a stable state is reached such that all the weight changes are zero; at this point, all the responses are either pinned near their minimum values or near $(\overline{y_j}/c_o)^p$. Some interesting results were reported for the development of tight orientation tuning curves for units, using simple oriented line patterns as inputs.

One problem with this approach is that the criterion of maximal selectivity cannot be directly optimized since it is not a continuous function. The authors experiment with a number of learning rules with different choices of the function Φ (e.g., continuous versus discontinuous, with different values of p and c_o) which satisfy the general form given above. Unfortunately, not all choices of Φ necessarily result in learning rules which converge to high-selectivity solutions. For example, choosing p close to zero and Φ continuous would result in a relatively flat selectivity curve.

It would be preferable to choose a well-behaved measure of selectivity which is continuous with respect to the network parameters and can directly be optimized by a straightforward method such as gradient descent. Intrator[15] has proposed an extension of the above work, using an objective function which is related to the *skewness* of the distribution. This causes a unit to discover projections of the data having bimodal distributions. Further, he shows how this objective, when applied to a group of units which inhibit each other, leads to the discovery of multiple features in the data.

Linsker[9–11] developed a slightly more complicated Hebbian rule for networks of units with spatially localized receptive fields, operating on *random* input patterns. In Linsker's model, each layer of linear units is arranged in a two-dimensional grid, and a unit in layer L is connected to N_M of the units in the preceding layer M, with the probability of a connection falling off as a Gaussian function of the distance from the L unit's centre. The following Hebbian learning rule is used:

$$\Delta w_{kj}{}^{\alpha} = c_1 + c_2(y_k{}^{M\alpha} - c_3{}^{M})(y_j{}^{L\alpha} - c_3{}^{L})$$

where the c_i's are constants, the L and M superscripts denote layers, and the α superscript denotes a particular training pattern. The above rule can be averaged over an ensemble of training patterns, assuming the time course of learning is much longer than the time interval of a pattern ensemble. A weight on the ith input line to a unit changes in proportion to its covariance Q_{ij} with every other input j to the same unit:

$$\dot{w}_{kj} = k_1 + \frac{1}{N_M}\sum_i (Q_{ij}{}^{L} + k_2)w_{ki}$$

where k_1 and k_2 are more constants. Linsker experiments with different choices of the four parameters k_1, k_2, N_M and r_M^2/r_L^2 (the ratio of the areas of receptive fields from layer L to M). By fixing these parameters, and limiting the values of weights to lie within the interval $[-1, 1]$, the evolution of the weights to a unit in the first layer can be simulated. With random inputs, the covariance between input lines is zero, and the parameters k_1 and k_2 can be chosen so that these weights become "all-excitatory" (i.e., they are pinned at their positive extrema). Once this layer of weights is fixed, the correlation function for layer 1 units can be computed. Whereas units in the input layer are uncorrelated, units in the first layer have overlapping receptive fields which may share some inputs. Hence, layer 1 units will be partially correlated, and this effect falls off exponentially as the distance between units' centers. Now the maturation process can be simulated for each successive layer L, by choosing values for the four parameters, computing Q^L, and then solving the ensemble-averaged equation for the weights to that layer.

Linsker[9] finds that with appropriate choices of the learning constants, units in the intermediate layers evolve a progressively more "Mexican hat" shaped receptive field, much like the "on-centre, off-surround" and "off-centre, on-surround" receptive fields found in the early levels of mammalian visual systems. When the process is repeated for several more layers, again with appropriate selections of the four parameters at each layer, cells can develop into orientation-specific, banded receptive fields, much like the simple cells in the primary visual cortex.[10] To explain these results, Linsker derives an energy function, in which his learning rule performs gradient descent. He reports that the minima of this energy function found by the method of simulated annealing correspond closely to those found in his network simulations. Yuille *et al.*[16] have shed further light on Linsker's results by analyzing a closely related energy function; they show analytically that for certain forms of spatial correlation functions (e.g., the Laplacian of a Gaussian with a small asymmetry), the minimum energy solutions correspond to oriented receptive

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

fields. Furthermore, the addition of lateral inhibition leads to oriented quadrature pairs.[16]

Linsker's work is important for several reasons. First, it shows how the simple learning rule proposed by Hebb can be applied in multiple stages of learning, giving rise to interesting classes of feature detectors. Second, it demonstrates a mechanism by which visual systems could evolve, given purely random input and appropriate architectural constraints (i.e., spatially localized receptive fields with a Gaussian connectivity distribution), and may account for some of the neurophysiological findings on early perceptual development. Barrow[17] has shown that the same principles could operate on patterned input; he obtains similar results to Linsker's using natural images (after low-pass filtering and Gaussian windowing) as training patterns, in a Hebbian network of units with lateral interactions. Miller, Keller and Stryker[18] have performed similar simulations with units having binocular inputs and local lateral excitatory connections; using a Hebbian learning rule with decay in this case leads to the formation of ocular dominance columns.

Linsker's model can be criticised on the basis of its limited computational power, since he uses an entirely linear system. Hence the linear transformation computed by successive layers in his networks could be combined into a single linear transformation, expressible by a single layer of weights. A more pragmatic problem with this work is the large number of parameters required to specify the behavior at each layer; each of these parameters must be hand-tuned to produce the desired receptive fields at the top layer. In his later work, we shall see that Linsker takes a more principled approach by beginning with a simple information-maximizing objective for unsupervised learning. Nonetheless, the former approach has great potential for future research in neurobiological modeling. Linsker has shown how an orientation-selective map could develop with simple layered Hebbian networks; with extensions to nonlinear networks with recurrent connections, the possibility of modeling the simultaneous self-organization of multiple feature maps (e.g., maps of color, orientation, direction of motion, etc.) remains intriguing.

4. Algorithms that Compute Principal Components

Many of the early models of unsupervised learning were based on *ad hoc* learning rules which were determined empirically to produce interesting behavior. A more principled approach is to first postulate an objective for the learning, and from that, derive the weight update equations; this provides a theoretical motivation for the model, and makes it easier to analyze the learning procedure and predict its behavior. One possible objective of unsupervised learning is to discover the principal components of the input distribution, or the N most significant components (i.e., the eigenvectors of the input correlation matrix corresponding to the N largest eigenvalues). Figure 1 illustrates the directions of principal variation for a two-dimensional distribution. In this section, we review several neural network learning algorithms which are related to Principal Components Analysis (PCA). These algorithms learn a set of linear orthogonal projections in the directions of principal variation in the input distribution, or a rotated subspace of these directions. Although PCA is a standard statistical technique which has been around for a long time, it would be of interest if such a transformation could be learned in an unsupervised neural network framework, particularly if an efficient parallel algorithm could be developed.

Oja[19] analyzed a version of the simple Hebbian learning rule which rescales each updated weight to maintain the Euclidean norm of a unit's weight vector equal to one:

$$w_{ji}(t) = \frac{w_{ji}(t-1) + \varepsilon y_i(t) y_j(t)}{\left\{ \sum_{k=1}^{n} [w_{jk}(t-1) + \varepsilon y_k(t) y_j(t)]^2 \right\}^{1/2}}.$$

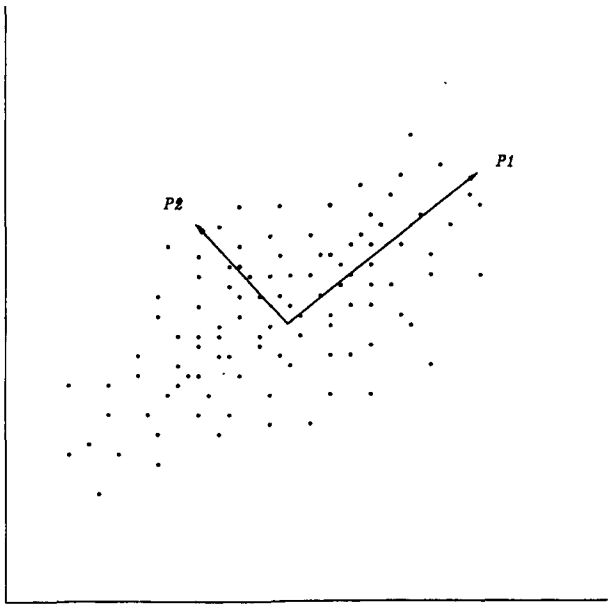

Fig. 1. A two-dimensional data distribution. The arrows, *P1* and *P2*, indicate the directions of principal variation.

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

For sufficiently small ε, this can be approximated by the following rule:

$$w_{ji}(t) = w_{ji}(t-1) + \varepsilon y_j(t)[y_i(t) - y_j(t)w_{ji}(t-1)] .$$

This rule causes each weight to grow in proportion to the product of the presynaptic and postsynaptic responses, as with the usual Hebbian rule, but adds an internal feedback term which limits the rate of growth in proportion to the size of the output on each case. Oja proved that this learning rule results in a unit that computes the first principal component of the input, i.e., its weight vector converges to (ignoring sign) the principal eigenvector $\mathbf{w}^*$ of the input correlation matrix, provided that the initial weight vector $\mathbf{w}(0)$ is not orthogonal to $\mathbf{w}$. Viewed another way, this rule results in a unit which maximizes the variance of its output, subject to the constraint that the norm of its weight vector is one.[20]

Extending this idea, Oja[21] considered a more complicated learning rule in which the internal feedback term for a particular unit depends not only on the output of that unit, but on a weighted combination of the activities of all the other units in the output layer:

$$w_{ji}(t) = w_{ji}(t-1) + \varepsilon y_j(t)\left[y_i(t) - \sum_k w_{ki}(t-1)y_k(t)\right] .$$

This rule says that a weight on a given connection should grow as the product of the presynaptic and postsynaptic responses, but that this effect should be reduced in proportion to the output values of other units which are connected to that same input unit, weighted by their connection strengths for that input. Oja called this type of learning network the Subspace Network, because it converges to a set of k weight vectors which span the same subspace as the coefficient vectors of the first k principal components of the input distribution.[21]

A related algorithm, GHA, was developed by Sanger,[22,23,20] which allows a group of linear units to learn an N-column weight matrix whose columns converge to precisely the first N principal eigenvectors of the input correlation matrix, in descending eigenvalue order. The GHA orthogonalizes the vectors using the sequential Gram–Schmidt algorithm: for each normalized weight vector w_j, subtract off the sum of its projections in the direction of the $j-1$ previous weight vectors, and finally, renormalize all the vectors. Combining Oja's normalized Hebb rule (which computes the first principal component) with Gram–Schmidt orthogonalization, GHA employs the following learning rule:

$$w_{ji}(t) = w_{ji}(t-1) + \varepsilon y_j(t)\left[y_i(t) - \sum_{k \le j} w_{ki}(t-1)y_k(t)\right] .$$

Note that this rule looks remarkably like Oja's subspace learning rule, except that in the GHA, the ith unit's weight updates depend on feedback only from the $j-1$ preceding units in the sequence (hence the inherent sequentiality of Sanger's algorithm), rather than *all* of the other units. So we can think of Oja's algorithm as creating a sort of "push and pull" process among all of the units to attain orthogonal weights while each tries to account for as much variance as possible; in contrast, Sanger's algorithm designates one particular unit to compute the first principal component by accounting for as much of the variance as possible, another to compute the second component by remaining orthogonal to the first unit and computing the projection that accounts for as much as it can of the remaining variance, another to compute the third (by remaining orthogonal to the first two), etc.

One problem with the GHA is that its method of finding successive eigenvectors is numerically poor, so it is best suited to finding only the first few eigenvectors of high-dimensional data.[20] On natural images, Sanger reports that these components are similar to the center-surround and oriented feature detectors discovered by Linsker's networks[9–11] and Barrow's networks.[17] This may limit its ability to model the full range of feature detectors (e.g., at varying spatial frequencies) found in primate visual systems; when applied to natural images, it "does not discover filters with different frequency response until very late in eigenvalue order, and it requires many iterations to produce poor-quality results."[20]

A general advantage of a "first N components" PCA or subspace decomposition is that it can be used to reconstruct the original inputs, and in fact it is the optimal linear transformation for N coefficients with respect to the mean-squared error of reconstruction (for a proof, see, e.g., Cottrell *et al.*[24]). Hence this approach to unsupervised learning is useful if the main objective is to achieve good data compression, while retaining as much information as possible about the

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

input. For this purpose, the first N components representation is preferable to an arbitrary subspace representation, because the principal components are ordered in terms of decreasing variance; hence we could obtain optimal compression by employing a variable length encoding, using the greatest precision to encode the first component, and progressively less precision to encode subsequent components. On the other hand, if units compute a function that has a limited dynamic range (e.g., if they use a sigmoidal activation function), it may be advantageous (w.r.t. information loss) for units to have roughly equal variance. Further, if the precision of units is limited due to some source of noise, it may be desirable that their outputs be correlated.

A deep criticism of PCA-related learning procedures for neural networks is the questionable utility of this type of processing for typical learning problems. The approach offers the advantage of data compression, if we transmit only the most significant components, but what good is this compressed representation? Its optimality with respect to accurate input reconstruction has questionable relevance for biological perceptual systems, as Linsker has pointed out.[25] We could very accurately reconstruct the inputs by simply copying them, thereby using very little hardware. In general, it appears that brains do much more than simply try to reproduce the inputs received by their sensors, but rather, some underyling "meaningful variables" or features are extracted which allow high-level interpretations. So a more relevant question may be how useful the PCA representation is for further perceptual processing.

A principal component analyzing network may be a useful preprocessor for later learning stages, since it has the nice property of finding *uncorrelated* projections. This will typically be very helpful for a method such as back-propagation learning by steepest descent. One thing that makes supervised back-propagation learning slow is that there may be interacting effects of different weights on the error. Because of these interactions, convergence can be expected to be very slow, even with simple local accelerating procedures such as momentum[26] or the use of adaptive learning rates for each weight (e.g., the "delta-bar-delta" algorithm[27]). If, on the other hand, the input representation uses uncorrelated components, then the Hessian of the error function is more nearly diagonal, so simple acceleration methods will permit a considerable speedup, by scaling the learning rates appropriately along each weight axis independently. (Note that a subspace method such as Oja's learns *orthogonal* but not necessarily *uncorrelated* projections so it does not offer this advantage).

In some cases, the principal components decomposition may coincide with features of interest in the input, as Sanger[20] has found. Oja[21] suggests that subspace methods are particularly useful for representing and classifying patterns such as spectra and histograms. For arbitrary input distributions, however, there is no guarantee that a subspace method or PCA will capture the interesting structure. PCA would be expected to represent poorly parameters which vary in a highly nonlinear manner, and may in fact, depending on the distribution, obscure clusters in the data (i.e., if there are too many isotropically distributed clusters[28]). In general, PCA appears to be a useful form of preprocessing for later unsupervised or supervised learning. Further empirical work needs to be done to determine the sorts of input distributions for which this standard linear transformation sufficiently captures the structure of interest, how many components must be computed to adequately represent data such as natural images, how the complexity of the learning compares to standard PCA methods, and to what extent this preprocessing speeds later learning.

5. The Encoder Learning Paradigm

Another approach to unsupervised learning is the "encoder paradigm." The goal is to store a collection of N patterns so that they be recalled as accurately as possible. Other potential benefits of the process are data compression, feature discovery, generalization to patterns which were not in the training set, and pattern completion (i.e., retrieval of one of the stored patterns when only part of it is supplied). Good pattern completion generally requires iterative retrieval, and is achieved to varying degrees by learning algorithms for recurrent networks such as the Hopfield network,[29] the Boltzmann machine,[30] and recurrent back-propagation nets,[31–33] by creating basins of attraction around the stored patterns in "state-space." In order to achieve good generalization, the hidden units of the network must discover interesting features of the input distribution. For this reason, the representation formed by the hidden layer of encoder networks is of primary interest in the context of our discussion on unsupervised learning.

Back-propagation can be used within the encoder paradigm to train a network to learn the identity mapping, by making the desired states of the N output units identical to the states of the N input units on each case.[34] We refer to this type of network as an

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

autoencoder (see Fig. 2). Typically some number of hidden units $M < N$ is used. By passing the input through this "bottleneck," some degree of data compression is achieved. The network can potentially learn to encode interesting features of the input patterns, by being forced to find a less redundant code with a small number of hidden units. Some generalization can therefore be expected, since patterns which were not in the training set should produce codes which are somewhere in between the codes of the closest patterns in the training set. Thus, as in supervised back-propagation, the network should learn some sort of smooth interpolation over the patterns in the training set. (Note that an autoencoder network would have limited pattern completion capabilities, since a partial input pattern would be expected to cause roughly the *average* of several stored patterns to be produced at the output layer, rather than the *nearest* stored pattern.)

Zipser[35] has shown that nonlinear autoencoder networks can discover interesting representations of image features, using very simple two-dimensional images of points blurred through a Gaussian, and then discretely sampled, as illustrated in Fig. 2. When each image consists of a single Gaussian spot in varying positions, and two hidden units are used to encode the ensemble of images, the hidden units form an orthogonal code for 2D position (which in this simple case is sufficient to accurately reconstruct the entire image). As the number of hidden units is increased, there is a gradual transition from a "variable code" in which spot location is a direct function of hidden unit activity, to a "value code" in which hidden units' responses over the space of 2D coordinates form complex, roughly oscillating receptive field patterns which overlap to varying degrees. Zipser[35] also applied the autoencoder to the "stereo" task of reproducing input patterns consisting of pairs of one-dimensional images of Gaussian spots at varying disparities. When four or more hidden units were used, there was some degree of disparity between the two dots in the reconstructed image. Interestingly, with four hidden units, rather than encoding the two input images independently by position, with two hidden units devoted to each image, a binocular or depth code was always learned by each hidden unit.

Finally, Zipser[35] used the representation learned by a position-encoding net as the input to another network that was trained to predict spot location when shown the first two images in a sequence of three images of a moving spot. The prediction network receives as input the hidden-unit representations learned by the monocular spot encoder for the first two images in the sequence, and the desired output is an image of the spot in the third position in the sequence. The network learns the appropriate transformation of the positions of the first two spots to encode the positions of the predicted spot in its hidden units. Although this is a simple linear transformation (i.e., an *easy* learning problem), it does nicely illustrate the idea that the representations discovered by an unsupervised learning procedure can be used beneficially by later learning stages.

Cottrell *et al.*[24] studied the effectiveness of nonlinear autoencoder networks in performing image compression. They trained networks to learn identity mappings on inputs consisting of small patches of a large image, with varying degrees of compression (achieved

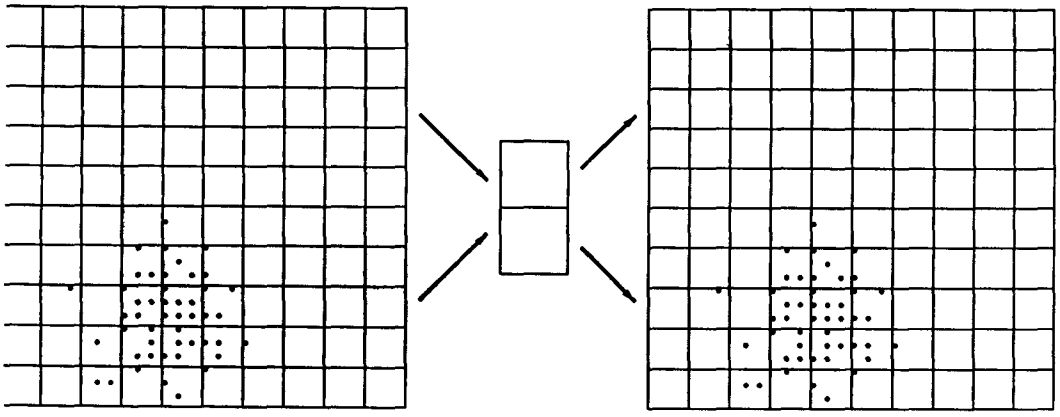

Fig. 2. The autoencoder architecture used by Zipser[35] to encode the 2D position of a Gaussian spot.

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

by decreasing the number of hidden units and by quantizing the hidden units' outputs by different amounts). One interesting finding is that hidden units tended to learn roughly equal variance encodings (clearly not a principal components decomposition). The authors suggested that this is due to the face that back-propagation tends to distribute the error fairly evenly to the hidden units. This might also account for Zipser's finding that hidden units in his stereo network always learned features common to both images rather than dividing themselves among the two images and learning separate encodings. A second interesting finding is that when the nonlinear back-propagation network was compared to a purely linear autoencoder, the latter gave much better generalization results (with respect to mean-squared reconstruction error) when applied to new images. The authors' explanation for this effect is that no quantization was applied to the outputs of the hidden units in the linear case, so they had a much larger dynamic range with which to represent images. A second possibility is that the network could have learned a nonlinear function which exactly fits (i.e., overfits) the training set, but generalizes much more poorly than would a smoother function.

An interesting open question remains: what advantages, if any, do nonlinear neural networks have over linear encoders (including PCA) with respect to data compression, quality of image reconstruction, and the nature of the learned representation?

In a purely linear network, the autoencoder paradigm is equivalent to PCA with respect to computational power, since a linear back-propagation autoencoder converges to a set of weight vectors which is a linear combination of the eigenvectors of the input correlation matrix.[36] This decomposition does not necessarily result in uncorrelated components, so in this respect, PCA or Sanger's GHA is superior. Furthermore, the autoencoder requires an extra layer of hardware through which gradients must be back-propagated, compared to GHA or Oja's subspace network.

In the nonlinear case, however, Zipser showed that the autoencoders could learn rather interesting representations, particularly as the number of hidden units was increased. With more units, a "value code" emerged in which units partitioned themselves over the space of the underlying input parameters (position and depth) in something like a coarse coding of the parameter space. It is unclear whether such easily interpreted representations would be learned for more realistic images with many underlying parameters, or whether it is even a reasonable goal to try to encode *everything* about the input distribution. Still, many interesting experiments remain to be done to study the nature of codes that can be learned by nonlinear autoencoders, and surprisingly little research has been published on this topic.

One interesting question is what sort of codes would be discovered if, instead of squeezing the input through a bottleneck, it was expanded into an even wider layer of hidden units. Barlow *et al.*[37] discuss the advantages of using such an "expansive code" in sensory systems. A trivial solution for the autoencoder in this case is to simply copy the inputs to a subset of units in the hidden layer, but Zipser's experiments suggest that an autoencoder might learn other solutions. Another question is what type of constraints could be imposed on the representations learned by nonlinear autoencoders to learn, for example, independent, mutually exclusive or orthogonal features. Finally, it would be interesting to compare autoencoders with the clustering algorithms which we discuss in the next section, in terms of their ability to model multimodal distributions.

6. Algorithms that Perform Clustering

Many algorithms have been developed for unsupervised learning in artificial neural networks which are variants of statistical clustering algorithms, and are generally referred to as competitive learning procedures.[38–42] The basic idea underlying competitive learning is to have some sort of competition between units' responses, so that only one unit in each competitive cluster tends to become active for each input pattern or pattern class. If a limit is imposed on the number of patterns each unit can respond to, units will tend to partition themselves among the input patterns fairly evenly.

One way to induce a winner-takes-all competition is via lateral inhibitory connections (i.e., negative links) between units within a layer; such a mechanism was originally proposed by Rosenblatt[43] in one of his perceptron models. In von der Malsburg's early competitive learning model,[38] a spatially localized version of this type of competition is produced using short-range lateral excitatory and inhibitory interconnections with fixed weights (arranged so as to produce a roughly Mexican-hat-shaped spatial correlation function). Adaptive weights from the input layer are trained with a simple Hebbian learning rule, with a renormalization term to force the weights of each unit to sum to a constant. When trained on simple binary

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

patterns consisting of oriented bars, units learn to respond preferentially to particular orientations. Furthermore, neighboring units tend to respond to similar orientations, while next-to-neighboring units respond to nearly perpendicular orientations, so that the response profile across the lattice of units forms an orientation map qualitatively similar to that seen in the visual cortex. Von der Malsburg's learning equation is similar to Oja's simple normalized Hebbian rule, except that they use different norms. The result is very different, however, because the recurrent connections cause units to tend to be positively correlated with their nearest neighbors, and negatively correlated with their more distant neighbors. When this effect is combined with a learning rule which causes units to discover correlations in the input, the result is that adjacent units learn maximally overlapping features, while units separated by short distances discover minimally overlapping features, which tend to be *mutually exclusive*.

Fukushima's Cognitron[39] is a multilayer version of competitive learning which performs a simple type of hierarchical clustering. The basic model is similar to von der Malsburg's; the main differences are a slightly more complicated multilayered architecture with probabilistically interconnected units, strictly inhibitory local connections within layers, and an explicit winner-takes-all learning rule: a unit only adapts its weights when it is the most strongly active in its local neighborhood. The receptive field radius increases with progressive layers, so that simple features of small spatial regions are learned in lower layers, and higher-order features (corresponding to larger spatial regions of the input) are learned by higher layers. On a very simple digit recognition task, at the fourth layer, most units learn to respond selectively to single digits. In the Neocognitron,[44] the model is extended to achieve shift-invariant pattern recognition, by replicating local feature detectors at multiple spatial positions within each level in the hierarchy. Ambros-Ingerson *et al.*[45] have also proposed a competitive learning algorithm for hierarchical clustering, in a model of the rat olfactory cortex.

Rumelhart and Zipser[46] have studied a simplified version of competitive learning which captures many of the essential features of the above models, but is much easier to analyze. Rather than implementing the winner-takes-all mechanism using lateral inhibition, they dispense with the recurrent links and simply use a nonlinear activation function which sets the activity of the winning unit (the one with the greatest total input) to one, and the rest to zero. They use the following learning rule:

$$\Delta w_{ji}{}^{\alpha} = \begin{cases} 0 & \text{if unit } j \text{ loses on pattern } \alpha \\ \varepsilon\left(\dfrac{y_i{}^{\alpha}}{\sum_k y_k{}^{\alpha}} - w_{ji}\right) & \text{if unit } j \text{ wins on pattern } \alpha. \end{cases}$$

By redistributing some proportion ε of each of the winning unit's weights to the weights on its active input lines, this rule maintains the constraint that $\Sigma_i w_{ji} = 1$, for each unit j. We can think of this learning rule as causing each unit to move its weight vector toward the mean of the cluster of patterns for which that unit responds. Hence, each unit is approximately performing gradient descent in the squared distance between its weight vector and the patterns in its cluster, which is equivalent to the standard k-means clustering algorithm.

Kohonen's model of unsupervised topological map formation[40,47] has much in common with the competitive learning models discussed so far. He uses a network of units arranged in an array (usually two-dimensional) such that there is some topological neighborhood, $N_i(t)$, defined for each unit i. The "winning unit" c is the one for which the Euclidean distance between its weight vector and the current input vector is minimal. Every unit within the winner's neighborhood, N_c, adapts its weights according to the following learning rule:

$$\Delta w_{ji}{}^{\alpha} = \begin{cases} \varepsilon(y_i{}^{\alpha} - w_{ji}) & \text{if unit } j \in N_c \\ 0 & \text{otherwise.} \end{cases}$$

If both the learning rate ε and the neighborhood size shrink gradually over the course of learning, the units' responses tend to become distributed evenly over the input probability distribution. For neighborhoods of size 1, Kohonen's algorithm is equivalent to k-means clustering.[47] For larger neighborhoods, the algorithm is a generalization of k-means which adapts each weight toward the centre of its own cluster of patterns *and* its neighbors' clusters, resulting in an ordered mapping that tends to preserve the topological structure of the input distribution. Kohonen has applied this algorithm to preprocessed speech data, and found that the clusters found by units usually correspond to phonemes. The sequences of these "quasi-phonemes" produced by processing a sequence of time slices of the speech signal can be viewed as an ordered trajectory through a "phonological map," indicating that the network has

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

learned to represent similar sounds at nearby locations in the map.

With a more elaborate architecture employing both feedforward and recurrent feedback connections, Carpenter and Grossberg's version of competitive learning[42,48,49] allows some degree of sequential matching, so that a new cluster will be formed only if a new pattern is sufficiently different from all of the clusters formed so far. By appropriately tuning the model parameters, one has control over the plasticity–stability tradeoff, so that the system tries to flexibly adapt to new classes of inputs without destroying the representations it has learned so far.

A further extension to the basic competitive learning model is to make each unit's response a *Gaussian* function of the distance between each input pattern and its weight vector. Each unit therefore becomes maximally tuned to some particular region of the input space, and this tuning falls off exponentially in all directions. Moody and Darken[50] applied this type of competitive learning as an unsupervised preprocessing stage to speed up subsequent supervised learning. They trained an adaptive layer of Gaussian units using the standard *k*-means clustering algorithm to adjust the Gaussian centres; an additional layer of units was then trained by a supervised learning procedure (mean-squared error minimization) to solve two difficult problems: phoneme recognition and chaotic time series prediction. They report that this hybrid algorithm results in about two orders of magnitude speedup in the learning time compared to pure supervised learning with back-propagation. The layer of Gaussian units speeds learning for two reasons: First, two input vectors which are far apart (in Euclidean distance) will tend to activate nonoverlapping sets of Gaussian units, so there will be no interference between these two training cases. Second, the incoming weights of Gaussian units do not depend on the outgoing weights so there is modularity of learning between these two layers of weights.

Recently, Nowlan[51] proposed a "soft competitive learning" method; rather than only allowing the winner (or winning neighborhood) to adapt, each unit can adapt its weight for every input case, in proportion to how strongly it responds on a given case. As a group the units form a good multimodal model of the underlying distribution, by performing gradient ascent in the model likelihood. The learning procedure is similar to Dempster's EM algorithm[52] for the special case of Gaussian mixture components. Nowlan showed that this method is superior to the traditional "hard competitive learning models" on two classification tasks, handwritten digit and vowel recognition (when the unsupervised learning for each algorithm is followed by a linear supervised layer). Furthermore, compared to a nonlinear multilayer back-propagation network on the digit task, the soft competitive model required roughly an order of magnitude fewer training examples to achieve comparable classification performance.

Clustering algorithms are useful when the goal is to find a limited number of prototypes in high-dimensional data. Each competitive unit or module comes to encode the center of gravity of a cloud of data points in a high-dimensional space. Whereas PCA may obscure clusters in some data distributions, in such cases, competitive learning schemes may be advantageous, since they attempt to represent explicitly the locations of clusters. Each cluster may come to represent implicitly conjunctions of interesting features, which can be interpreted as describing prototypical class members. However, there are well-known problems with standard clustering algorithms which are shared by competitive learning. The effectiveness of such learning schemes tends to be very dependent on the number and distribution of clusters in the data, hence, the architecture must be carefully chosen to suit the input distribution. If a measure such as Euclidean distance is used for within-cluster similarity, the algorithm will perform best on data having compact, well-separated clouds of points[53]; nonspherical or more scattered natural clusters will tend to be broken up into smaller ones by the algorithm, and outliers will be poorly represented. Finally, most competitive learning algorithms based on a hard "winner-takes-all" strategy are very sensitive to the initial conditions (i.e. initial random weights), and the order of presentation of input patterns.

If later learning requires interpolating a smooth function from points in the input distribution, then cluster centers, or more generally any set of radial basis functions, will make a good set of interpolation points.[54] However, in some situations there are disadvantages to this type of representation. We are "spreading out" the input distribution into a higher-dimensional space, and effectively partitioning it into a number of (more or less) disjoint subregions of feature space, as illustrated in Fig. 3. In doing so, we may scatter essential information about the continuity of individual features, and about low-dimensional combinations of continuously varying features. It is therefore difficult to make a multilayer version of this type of unsupervised learning algorithm. We would like an unsupervised learning procedure to be applicable at

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

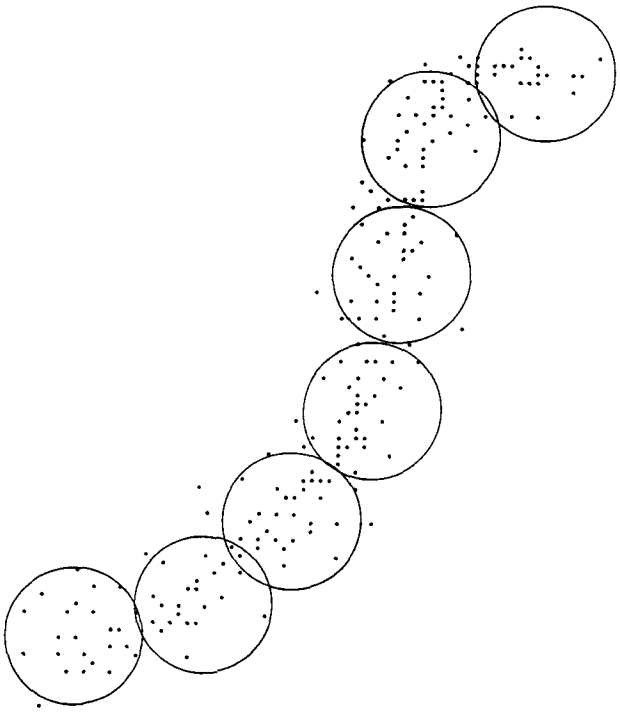

Fig. 3. A two-dimensional data distribution, with circles illustrating the locations where a typical clustering algorithm (e.g., *k*-means or competitive learning) would place the cluster centers.

successive layers, so that it can extract and explicitly represent progressively higher-order features. Finally, it would be interesting if a (nonhierarchical) clustering algorithm could be developed which discovers structure at multiple scales.

7. Algorithms that Maximize Information Transmission

Several of the papers discussed so far have viewed the goal of unsupervised learning as finding an optimal encoding of the input patterns. For example, it has been pointed out that PCA is the optimal linear encoding with respect to mean-squared reconstruction error.[22,23,20,24]

An alternative optimality criterion proposed by Barlow[55,56] is to find an encoding which is minimally *redundant*. Barlow suggests that the goal of unsupervised learning should be to find an encoding of the input which makes it easy to form new associations between sensory events and reward/punishment. If the encoding of the sensory input vector into an n-bit feature vector has the property that the n bits are statistically independent, then all that is required to form new associations with an event V is knowledge of the conditional probabilities $p(V|y_i)$, for each feature y_i (rather than complete knowledge of the probabilities of events conditional upon each of the 2^n possible sensory inputs). Barlow proposes that one way to achieve bitwise independence is to find a *minimum entropy encoding*: an invertible code (i.e., one with no information loss) which minimizes the sum of the bit entropies. Although this permits high "within-bit redundancy" (i.e., the expected values of individual bits may deviate from 0.5), it minimizes the "between-bit redundancy" (the extent to which one feature is predictable from some combination of the other features). Barlow's minimum entropy objective intuitively seems to capture the desirable characteristics of a sensory processing system; unfortunately he does not propose any direct method of finding such a code. A simpler objective of learning *decorrelated* codes via an anti-Hebbian learning rule has been proposed by Barlow and Földiák.[57] Some serial heuristic search algorithms for finding minimum entropy codes are described by Barlow *et al.*[37] Földiák[58] has shown how a network of units with feedforward connections trained by a normalized Hebb rule, and lateral feedback connections trained by an anti-Hebbian rule, can learn a sparse code which approximates Barlow's objective, by reducing the statistical dependency between features while preserving most of the information about the input patterns.

An objective similar to Barlow's, proposed by Pearlmutter and Hinton,[59] is to try to discover features which account for redundancy in the sensory input. The Gmax algorithm[59] causes a unit to discover statistical dependencies between its input lines by maximizing the difference between the output distribution of the unit, P, in response to structured input, and the distribution, Q, that would be expected if the input lines were independent. Using probabilistic binary units, the Gmax algorithm maximizes the asymmetric divergence between these two distributions:

$$G = P \log \frac{P}{Q} + (1 - P) \log \frac{1 - P}{1 - Q}.$$

When a unit is trained on images of oriented bars, it learns center-surround receptive fields much like those learned by Linsker's Hebbian network.

The G-error is a very powerful objective, and can potentially capture arbitrarily high-order structure in the input distribution. It would be interesting to apply

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

the Gmax learning procedure to an output unit with several hidden units, using back-propagation of the G gradient to train the hidden units, to see if it could learn higher-order features which involve nonlinear combinations of input values, and which could not be learned by one-layer feedforward networks.

Unfortunately, there is no straightforward generalization of the Gmax principle to multiple output units. Pearlmutter and Hinton propose two possible mechanisms: adding an extra term to the objective function which minimizes the correlation coefficient between the outputs of units, and a mechanism of mutual inhibition. The former would encourage units to learn statistically independent features, whereas the latter would encourage the discovery of mutually exclusive features.

Recently, Linsker[60] has proposed an information-theoretic objective for perceptual processing which he calls the *Infomax* principle: a network should learn a mapping which preserves as much information as possible about the input vector. Linsker analyzes the consequences of this principle for two special cases. In the first case, the output of a unit is a linear function of its total input plus a noise term n. Both the input and noise are assumed to have Gaussian distributions. For this case, the rate at which the unit's output y_j transmits information about its input is:

$$R = 0.5 \ln\left[\frac{V(y_j)}{V(n)}\right]$$

where $V(n)$ is the variance of the noise. For this model, assuming the noise variance is fixed, maximizing R is equivalent to maximizing the variance of the unit's output. (Note that this is equivalent to Hebbian learning, as discussed in Sec. 4.)

In the second model Linsker considers, there is Gaussian noise n_i of variance $V(n)$ added to each (Gaussian) input line i. Now the information rate is:

$$R = 0.5 \ln\left(\frac{V(y_j)}{V(n)\sum_i w_i^2}\right).$$

For this model, the maximum information rate is achieved when the output variance is maximized while the length of the weight vector is minimized. In this case, we have a principal-component analyzing unit.

The situation becomes more interesting for models with multiple units. Plumbley and Fallside[61] analyze the case of a linear network with additive Gaussian noise which performs dimensionality reduction. They note that the information loss in the mapping from the inputs to the outputs is bounded from above by the entropy of the error in reconstructing the inputs. This entropy can be minimized by minimizing the mean-squared reconstruction error. So in this case, the optimal encoding with n units is the first n principal components. Linsker[60] also analyzes this case; he points out that the information rate for a collection of linear units with Gaussian noise is:

$$R = 0.5 \ln\left[\frac{\mathrm{Det}(Q^y)}{V(n)}\right]$$

where $\mathrm{Det}(Q^y)$ is the determinant of the covariance matrix of the output vector y. This results in a tradeoff between maximizing the variances of the outputs and decorrelating them. If the noise variance is large, the latter term predominates in the calculation of the determinant, and some redundancy is therefore desirable in increasing the signal-to-noise ratio. In the absence of noise, assuming there is some dimensionality reduction, the optimal solution is a decorrelated set of outputs having maximal variance. Note that Barlow's minimum redundancy principle achieves the same end as Linsker's Infomax principle only in the latter case; as the noise level increases, the Infomax principle becomes one of *maximal* ("between-unit") redundancy.

Atick and Redlich[62] explore an extension of Barlow's minimum redundancy principle which applies to noisy channels. The model they study is of a perceptual system which receives as input $x = s + n_1$, a noisy (i.e., quantized) version of some underlying signal s. The goal of the perceptual system is to find a mapping into a (slightly noisier) vector, $y = Ax + n_2$ (where A is a linear matrix operator), which minimizes the following redundancy measure:

$$R = 1 - \frac{I(y; s)}{C(y)}$$

[where $I(y; s) = H(y) + H(s) - H(y, s)$ is the mutual information between y and s, and C is the channel capacity] subject to the constraint of no information loss from x to y:

$$I(y; s) = I(x + n_2; s).$$

It is assumed that $C(y) >= I(x; s)$, i.e., there is no dimensionality reduction in the mapping from x to y. Here, $C(y)$ is taken to be the maximum of $I(y; s)$

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

when the power in the output signals $\langle y_i^2 \rangle$ is held constant and there is no noise apart from quantization. In this case, if the output signal and noise have Gaussian probability distributions, the channel capacity has the following form:

$$C(y) = 0.5 \log \left\{ \frac{\text{Det}[\text{Diag}(Q^y)]}{\text{Det}[\text{Diag}(Q^\delta)]} \right\}$$

where $\text{Diag}(Q^y)$ is the matrix consisting of the diagonal elements of the covariance matrix of y, and zeroes elsewhere, and δ is the quantization noise. To obtain an explicit solution which minimizes the above redundancy measure, they propose using a Lagrange multiplier to implement the constraint of zero information loss, and minimize the following "generalized redundancy measure":

$$R' = C(y) - \lambda[I(y; s) - I(x + n_2; s)] .$$

In a noise-free system, the redundancy can be squeezed to zero by minimizing the diagonal terms of Q^y, thereby lowering the channel capacity (as defined above). This is equivalent to Barlow's minimum redundancy principle (for the special case of a linear system with Gaussian variables), which says that we should minimize the sum of the component entropies of y in order to obtain uncorrelated components. In a noisy system, however, much of the channel capacity is wasted by transmitting noise; the signal-to-noise ratio is improved in this situation by allowing correlated output components.

The difference between Atick and Redlich's generalized redundancy measure and Linsker's Infomax principle is that when the output has the same dimensionality as the input, the latter would yield infinitely many equivalent solutions: maximizing information transmission (i.e., the determinant of Q^y) can be achieved by any linear transformation $y = Ax$ such that the output spans the same space as the input. On the other hand, redundancy can be lowered in two ways (depending on the level of noise): by increasing $I(y; s)$, and by keeping I constant while decreasing C, thus decorrelating the outputs.

8. Spatial Coherence as an Objective Function

The algorithms described in the previous section attempt to learn as much information as possible about the input by maximizing information transmission through a network. An alternative is to attempt to constrain the learning problem by restricting the features of interest, in some way, to those useful for further perceptual processing.

Becker and Hinton[63,64] proposed that a good objective for perceptual learning is to extract higher-order features that exhibit simple forms of coherence across space or time. Using a number of network modules receiving input from nearby image patches, each module learns to discover higher-order parameters exhibiting spatial coherence by maximizing the mutual information between the variables extracted by neighboring modules. The authors initially experimented with modules of binary stochastic units, on simple one-dimensional binary random dot stereograms. By performing gradient ascent in this information measure, units learned to discriminate between stereo patterns with different integer disparities.

They also proposed an algorithm for training modules of real-valued units to extract disparity from stereograms of surfaces with continuously varying depths. In this case, the underlying depth distribution was assumed to be Gaussian. The simplest model they studied applies to fronto-parallel planar surfaces. Since the depth is constant between adjacent patches in a given image, two modules A and B that receive input from neighboring patches should learn to compute a function such that their outputs a and b are approximately equal. Module A views b as a signal to be predicted, and its own output a as a noisy version of that signal. Under this model, the information that a provides about b is determined by the log ratio of two variances:

$$I_{a;b} = 0.5 \log \frac{V(\text{signal} + \text{noise})}{V(\text{noise})}$$

$$= 0.5 \log \frac{V(a)}{V(a - b)}$$

which is the standard expression for the information transmission rate through a noisy channel when the signal and noise have Gaussian distribution.[65] Using this information measure, when equality constraints are imposed on the weights so that each module is constrained to compute the same function, the output units of modules learn to encode real-valued depths from random dot stereograms of fronto-parallel surfaces.[63]

The second model studied by Becker and Hinton applies to curved surfaces, in which the parameter extracted at one patch is no longer equal to that of neighboring patches, but is now some unknown combination of parameters extracted in the neighborhood

region. The unknown coefficients of this linear function are learned by optimizing the same information rate, $I(a; b)$, where a is the predicted signal in a patch, and now b is a linear combination of the signals extracted by neighboring patches, as shown in Fig. 4. This model was used to train modules to interpolate continuously varying depths in curved surfaces. Units learned to accurately predict depth extracted in one image region by receiving input only from the surrounding region.

Atick and Redlich[66] have proposed a model of the development of retinal ganglion cell kernels (which have a center-surround response profile) which is similar to the above interpolation model, and also assumes a form of spatial coherence. They propose two objectives for ganglion cell development, which minimize redundancy and information loss under different levels of the signal-to-noise ratio. When the signal-to-noise ratio is high, the input signal at one retinal pixel location m, $x[m]$, can typically be predicted by some value $\hat{x}$ which is a linear combination of the neighboring signals: $\hat{w}[m] = \sum_{n\neq 0} a[n]x[m-n]$. The information content of each component of the transformed signal y can be increased (assuming some degree of spatial correlation) by only transmitting the amount by which the original signal differs from the predicted signal, $y[m] = x[m] - \hat{x}[m]$. The authors analytically derive an optimal prediction kernel (with respect to mean-squared error) a, for the special case when the (one-dimensional) input signal has an exponentially decreasing spatial correlation function. This kernel is a center-surround function which closely approximates the response profile of retinal ganglion cells. An unsupervised algorithm is proposed, which allows a unit to learn this linear predictor, using a simple negated Hebbian learning rule. For high levels of noise, redundancy reduction is less important and information loss is a more critical issue; in this case, a better way to transmit the input signal is by a *smoothing* kernel which can be learned by a *supervised* procedure.

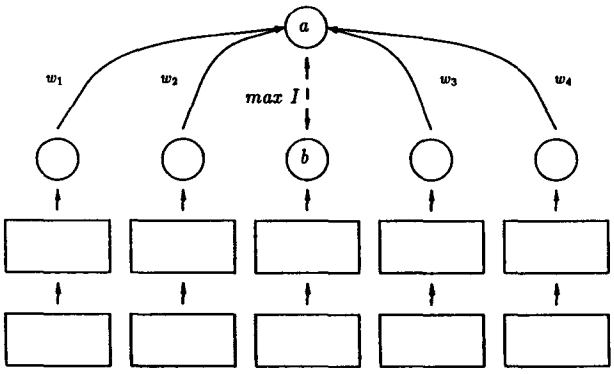

Fig. 4. A network used by Becker and Hinton[63] in which the goal of the learning is to maximize the information, $I(a; b)$, between the output of a local module (b) and the contextually predicted output (a) that is computed as a linear combination of the outputs of nearby modules.

The linear prediction operator proposed by Atick and Redlich is similar to the depth interpolation operator proposed by Becker and Hinton, but there are two main differences. First, in the Atick and Redlich model, each unit transmits the prediction *error* rather than the predicted signal, so the unit becomes a type of *invariance detector*; this is a more efficient way of transmitting information about the input signal. Second, Atick and Redlich used mean-squared error as the prediction criterion (assuming a fixed input signal), whereas Becker and Hinton use the mutual information (with Gaussian assumptions) between the original and the predicted signals. The latter objective can be applied to multilayer nonlinear networks, and discovers features which have high entropy, in addition to being highly predictive (whereas minimizing squared error in the multilayer case could produce low entropy, trivial solutions, e.g., with every weight equal to zero).

The models studied by Becker and Hinton apply to modules that have each a single real-valued output. For modules with several real-valued outputs, the natural way to generalize the mutual information measure between variables is to use the information rate for multidimensional Gaussians, where the variance is replaced by the determinant of the covariance matrix.[65] Zemel[67] has applied this algorithm to the problem of extracting the parameters of the viewing transform in images of simple two-dimensional objects.

There are many other ways of extending this work. For example, a better depth interpolator could be learned by a network with recurrent connections; this type of network could interpolate across regions in the input where no information was available, or depth estimates were noisy, since the network could relax into a consistent model. Another extension would be to learn temporal regularities in moving images, speech signals, etc; Becker and Hinton have used coherence across space, but the same techniques could be applied to coherence across time.

9. Conclusions

We have reviewed several classes of algorithms for unsupervised learning. The early work on Hebbian

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com
by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

learning by Bienenstock *et al.*[7] and Linsker[9–11] was concerned with designing variations of the basic Hebbian modification principle which would learn interesting perceptual features. The limitations of this approach are the lack of well-defined objective functions, resulting in complex learning rules with many tunable parameters and the restriction to linear systems. Intrator has suggested an objective function which would implement Bienenstock's principle of maximal selectivity in nonlinear networks. Linsker,[60] in his later work, proposed the Infomax principle as an objective for perceptual learning.

We also considered algorithms which compute some (possibly rotated) subset of the first few principal components of the input distribution.[19–23] This is an optimal linear transformation with respect to mean-squared reconstruction error; on the other hand, the relevance of this criterion for biological processing systems is questionable. PCA may be a useful preprocessing stage to speed later learning; however, such a representation has its limitations, depending on the nature of the input distribution. An open question is the extent to which any linear transformation of sensory input can capture the important features of the world. There is room for further empirical work to determine how many components are required to model data such as natural images or speech signals, and to what extent this type of processing facilitates (or detracts from) later learning of higher-order nonlinear features.

An interesting generalization of PCA algorithms is the autoencoder, a back-propagation network which learns the identity mapping.[34] In the linear case the autoencoder is equivalent to subspace methods, but in the nonlinear case, more interesting mappings can be learned.[35,24] More research needs to be done to determine how much this nonlinearity buys us over the linear PCA methods in accurately modeling the underlying distribution. One could further explore the nature of representations which can be learned by autoencoders, for example, when there are more hidden units than inputs, or when additional constraints are added to cause statistically independent or mutually exclusive features to be learned.

A number of unsupervised algorithms have been developed which perform clustering.[38–42,51] These methods are able to model structure in certain data distributions which would not be captured by the principal components. This kind of a representation may be a good solution for some classification problems, and is generally a useful preprocessing stage for interpolation problems. However, it is less desirable if the goal is to learn explicit representations of higher-order features. Future research is needed to develop learning methods which discover structure at multiple scales (and not in a strictly hierarchical manner, for there may be qualitatively different kinds of structure in the data), and can separate nonlinearly related underlying parameters. It may be possible to combine a subspace method, which discovers an orthogonal basis of features, with a clustering method, which breaks up the data along each dimension based on finer structure.

Several authors have proposed that a good objective for unsupervised learning is to maximize some measure of the information transmitted about the input. To get interesting solutions, one can add the constraint of dimensionality reduction, as Linsker[60] does, or use Barlow's principle of minimal redundancy[55,56] which encourages decorrelated output signals. In a Gaussian linear system, maximizing information transmission is equivalent to finding the principal components of the input distribution. We have questioned the utility of such a linear mapping for perceptual processing. In the general non-Gaussian and nonlinear cases, it is more difficult to obtain an explicit computational form for this objective. An interesting application of such an extension would be to maximize *temporal* information transmission through a recurrent network.

Rather than trying to transmit the entire information content of the input, one way to restrict the kinds of features the algorithms may learn is to optimize more constrained information measures such as spatial or temporal coherence. Atick and Redlich[66] showed that units can learn spatially invariant features of images by minimizing spatial prediction error. Becker and Hinton[63,64] showed that the mutual information objective can be used to train networks to discover spatially coherent features such as depth from stereo images. Several possible extensions of this work have been discussed. It remains to be demonstrated that the same techniques could be applied to the temporal domain.

Acknowledgements

I thank Geoff Hinton, Peter Földiák, Dave Plaut, Greg Dudek, Toni Pitassi, and the members of the Connectionist Research Group of the University of Toronto, for many helpful comments.

References

1. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations*

in the Microstructure of Cognition, volume I (Bradford Books, Cambridge, MA, 1986).
2. T. J. Sejnowski and C. R. Rosenberg, "Parallel networks that learn to pronounce English text," *Complex Systems* **1**, 145–168 (1987).
3. K. J. Lang and G. E. Hinton, "A time-delay neural network architecture for speech recognition," Technical Report CMU-CS-88-152 (Carnegie-Mellon University, 1988).
4. Y. le Cun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems 2* (Morgan Kaufmann Publishers, 1990) 396–404.
5. D. O. Hebb., *The Organization of Behavior* (Wiley, New York, 1949).
6. R. S. Sutton and A. G. Barto, "Toward a modern theory of adaptive networks: Expectation and prediction," *Psychol. Rev.* **88**, 135–170 (1981).
7. E. L. Bienenstock, L. N. Cooper, and P. W. Munro, "Theory for the development of neuron selectivity; orientation specificity and binocular interaction in visual cortex," *Neurosci.* **2**, 32–48 (1982).
8. B. Kosko, "Differential hebbian learning," in *Neural Networks for Computing*, ed. J. S. Denker AIP Conference Proceedings, Snowbird, Utah, 1986, 277–282.
9. R. Linsker, "From basic network principles to neural architecture: Emergence of spatial opponent cells," *Proc. Natl. Acad. Sci. USA* **83**, 7508–7512 (1986).
10. R. Linsker, "From basic network principles to neural architecture: Emergence of orientation-selective cells," *Proc. Natl. Acad. Sci. USA* **83**, 8390–8394 (1986).
11. R. Linsker, "From basic network principles to neural architecture: Emergence of orientation columns," *Proc. Natl. Acad. Sci. USA* **83**, 8779–8783 (1986).
12. G. Tesauro, "Simple neural models of classical conditioning," *Biol. Cybern.* **55**, 187–200 (1986).
13. A. H. Klopf, "Drive-reinforcement learning: a real-time learning mechanism for unsupervised learning," in *IEEE First Annual Conference on Neural Networks*, San Diego, California, 1987.
14. T. J. Sejnowski and G. J. Tesauro, "The Hebb rule for synaptic plasticity: Implementations and applications," in *Neural Models of Plasticity* (Academic Press, San Diego, 1989) 94–103.
15. N. Intrator, "A neural network for feature extraction," in *Advances in Neural Information Processing Systems 2* (Morgan Kaufmann, 1990) 719–726.
16. A. Yuille, D. Kammen, and D. Cohen, "Quadrature and the development of orientation selective cortical cells by Hebb rules," *Biol. Cybern.* **61**, 183–194 (1989).
17. H. G. Barrow, "Learning receptive fields," in *Proc. IEEE First Annual Conference on Neural Networks*, June 1987, 115–121.
18. K. D. Miller, J. B. Keller, and M. P. Stryker, "Ocular dominance column development: Analysis and simulation," *Science* **245**, 605–615 (1989).
19. E. Oja, "A simplified neuron model as a principal component analyzer," *J. Math. Biol.* **15**, 267–273 (1982).
20. T. D. Sanger, "Optimal unsupervised learning in feedforward neural networks," M. Sc. Thesis, Department of Electrical Engineering and Computer Science, MIT, 1989.
21. E. Oja, "Neural networks, principal components, and subspaces," *Int. J. Neural Systems* **1**(1), 61–68 (1989).
22. T. D. Sanger, "An optimality principle for unsupervised learning," in *Advances in Neural Information Processing Systems 1* (Morgan Kaufmann, 1989) 11–19.
23. T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforwad neural network," *Neural Networks* **2**, 459–473 (1989).
24. G. W. Cottrell, P. Munro, and D. Zipser, "Image compression by back-propagation: An example of extensional programming," ICS Report 8702, February 1987.
25. R. Linsker, "Designing a sensory processing system: What can be learned from principal components analysis?," Technical Report RC14983 (#66896), IBM, 1989.
26. D. C. Plaut, S. J. Nowlan, and G. E Hinton, "Experiments on learning by back-propagation," Technical Report CMU-CS-86-126, Carnegie-Mellon University, Pittsburgh PA 15213, June 1986.
27. R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," Department of Computer and Information Sciences COINS-TR-87-117, University of Massachusetts, Amherst, Ma., November 1987.
28. P. J. Huber, "Projection pursuit," *The Annals of Statistics* **13**(2) (1985) 435–475.
29. J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. USA* **79** (1982) 2554–2558.
30. G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing: Explorations in the microstructure of Cognition*, volume I, (MIT Press, Cambridge, MA 1986) 282–317.
31. L. B. Almeida, "A learning rule for asynchronous perceptrons with feedback in a combinatorial environment," in *Proc. 1st First International Conference on Neural Networks*, volume 2, San Diego, CA, June 1987 (IEEE) 609–618.
32. F. J. Pineda, "Generalization of back propagation to recurrent and higher order neural networks," in *Proc. IEEE Conference on Neural Information Processing Systems*, Denver, Colorado, November 1987 (IEEE).
33. P. Y. Simard, M. B. Ottaway, and D. H. Ballard, "Fixed point analysis for recurrent networks," in *Proc. Neural Information Processing Systems Conference*, Denver, December, 1988 (University of Rochester, 1988).
34. G. E. Hinton, "Connectionist learning procedures," Technical Report CS-87-115, Department of Computer Science, Carnegie-Mellon University, 1987.
35. D. Zipser, "Programming neural nets to do spatial computations," ICS Report 8608, June 1986.
36. P. Baldi and K. Hornik, "Neural networks and principal components analysis: Learning from examples without local minima," *Neural Networks* **2**, 53–58 (1989).
37. H. Barlow, T. P. Kaushal, and G. J. Mitchison, "Finding minimum entropy codes," *Neural Computation* **1**, 412–423 (1989).

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.

38. C. von der Malsburg, "Self-organization of orientation sensitive cells in striate cortex," *Kybernetik* **14**, 85–100 (1973).
39. K. Fukushima, "Cognition: A self-organizing multilayered neural network," *Biol. Cybern.* **20**, 121–136 (1975).
40. T. Kohonen, "Clustering, taxonomy, and topological maps of patterns," in *Proc. Sixth International Conference on Pattern Recognition*, ed. M. Lang, Silver Spring, MD, 1982 (IEEE Computer Society Press).
41. D. E. Rumelhart and D. Zipser, "Feature discovery by competitive learning," in *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*, volume I. (Bradford Books, Cambridge, MA, 1986).
42. G. A Carpenter and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer Vision* **37**, 54–115 (1983).
43. F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.* **65**, 386–408 (1958).
44. K. Fukushima and S. Miyake, "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position," *Pattern Recognition* **15**, 455–469 (1982).
45. J. Ambros-Ingerson, R. Granger, and G. Lynch, "Simulation of paleocortex performs hierarchical clustering," *Science* **247**, 1344–1348 (1990).
46. D. E. Rumelhart and D. Zipser, "Competitive learning," *Cognitive Sci.* **9**, 75–112 (1985).
47. T. Kohonen, "The 'neural' phonetic typewriter," *IEEE Computer*, **21**, 11–22 (1988).
48. S. Grossberg, "Competitive learning: From interactive activation to adaptive resonance," *Cognitive Sci.* **11**, 23–63 (1987).
49. G. Carpenter and S. Grossberg, "ART 2: Self-organization of stable category recognition codes for analog input patterns," *Appl. Optics*, 1987. Special issue on neural networks.
50. J. Moody and C. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation* **1**(2), 281–294 (1989).
51. S. J. Nowlan, "Maximum likelihood competitive learning," in *Advances in Neural Information Processing Systems 2* (Morgan Kaufmann, 1990) 574–582.
52. A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Proc. Roy. Stat. Soc.* **B39**, 1–38 (1977).
53. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis* (Wiley and Sons, 1973).
54. T. Poggio, "A theory of networks for approximation and learning," A. I. Memo No. 1140, C.B.I.P. Paper No. 31, MIT Artificial Intelligence Laboratory, and Center for Biological Information Processing, Whitaker College, 1989.
55. H. B. Barlow, "Cognitronics: Methods for acquiring and holding cognitive knowledge," Unpublished manuscript, October 1985.
56. H. B. Barlow, "Unsupervised learning," *Neural Computation* **1**, 295–311 (1989).
57. H. B. Barlow and P. Földiák, "Adaptation and decorrelation in the cortex," in *The Computing Neuron* (Addison-Wesley Publishing Corp., 1989) chapter 4, 54–72.
58. F. Földiák, "Forming sparse representations by local anti-hebbian learning," *Biol. Cybern.* **64**, 165–170 (1990).
59. B. A. Pearlmutter and G. E. Hinton, "G-maximization: An unsupervised learning procedure for discovering regularities," in *Neural Networks for Computing: American Institute of Physics Conference Proceedings 151*, ed. J. S. Denker, 1986, 333–338.
60. R. Linsker, "Self-organization in a perceptual network," *IEEE Computer* **21**, 105–117 (1988).
61. M. D. Plumbley and F. Fallside, "An information-theoretic approach to unsupervised connectionist models," *Proc. 1988 Connectionist Models Summer School*, 1988.
62. J. J. Atick and A. N. Redlich, "Towards a theory of early visual processing," Technical Report IASSNS-HEP-90/10, Institute for Advanced Study, Princeton, 1990.
63. S. Becker and G. E. Hinton, "Using spatial coherence as an internal teacher for a neural network," in *Advances in Back-Propagation*, ed. D. E. Rumelhart and Y. Chauvin, 1990 (in press).
64. G. E. Hinton and S. Becker, "An unsupervised learning procedure that discovers surfaces in random-dot stereograms," in *Proc. International Joint Conference on Neural Networks*, volume 1, Hillsdale, NJ, 1990 (Erlbaum) 218–222.
65. C. E. Shannon, "A mathematical theory of communication. Part iii," *Bell System Technical Journal* **XXVIII**, 623–656 (1949).
66. J. J. Atick and A. N. Redlich, "Predicting ganglion and simple cell receptive field organizations from information theory," Technical Report IASSNS-HEP-89/55, Institute for Advanced Study, Princeton, 1989.
67. R. S. Zemel and G. E. Hinton, "Discovering and using the single viewpoint constraint," to appear in *Advances in Neural Information Processing Systems 3* (Morgan Kaufmann Publishers, 1991).

Int. J. Neur. Syst. 1991.02:17-33. Downloaded from www.worldscientific.com by UNIVERSITY OF TORONTO on 02/24/14. For personal use only.