In Network: Computation in Neural Systems, Volume 7, Number 1, February 1996

Mutual Information Maximization: Models of Cortical Self-Organization

Suzanna Becker
Department of Psychology
McMaster University

Abstract

Unsupervised learning procedures based on Hebbian principles have been successful at modelling low level feature extraction, but are insufficient for learning to recognize higher order features and complex objects. In this paper we explore a class of unsupervised learning algorithms called Imax [1] that are derived from information-theoretic principles. The Imax algorithms are based on the idea of maximizing the mutual information between the outputs of different network modules, and are capable of extracting higher order features from data. They are therefore well suited to modelling intermediate to high level perceptual processing stages. We substantiate this claim with some novel results for two signal classification problems, as well as by reviewing some previously published results, and several related approaches. Finally, Imax is evaluated with respect to computational costs and biological plausibility.

1 Introduction

One approach to unravelling the learning rules employed by the brain is to start with neurophysiological data, and from there derive weight update equations. This "bottom-up" approach allows one to incorporate considerable biological detail in modelling synaptic level changes. For example, such an approach has led to refinements in Hebb's learning rule using recent evidence about conditions for LTP induction (e.g. [2]). However, in order to model the development of large-scale neural networks, whose behavior is not predictable from single neuron or synaptic-level events, we need an alternative approach. One such alternative is to work from the top down, beginning with a computational goal or cost function for the learning based on information processing constraints. This permits networks to be understood in terms of their *global* behaviour. Such an understanding can be elusive when one starts with a synaptic learning rule.

For example, the back-propagation learning procedure [3] minimizes the *mean squared error* between the network's output and the training signal. However, back-propagation learning is not considered biologically plausible, because for each training pattern, each output neuron must be provided with a target state. Further, error signals must be propagated backwards along connections. In unsupervised learning, there is no training signal. The problem in this case is to formulate an objective function for the learning that measures how well the network has encoded the information available to its sensors, independently of any direct external feedback to units. Once a cost function has been formulated, weight update equations can be obtained by differentiating the cost with respect to the weights. By performing gradient descent in a cost function we can thereby reduce a global algorithm into synaptic-level steps (weight changes). Note, however, that the converse is not necessarily true. That is, a given synaptic learning rule may not correspond to the derivative of any global cost function. For example, for a single neuron, Hebb's learning rule can be shown to follow the derivative of the output variance [4], however, in a multi-layer feed-forward network with fully interconnected layers, the variance of an output unit's activity depends on the weights to all units in the preceding layers. Thus, the Hebb rule can only be said to maximize a local cost function. It tells us nothing about the behavior of an entire network of units with Hebbian synapses.

Frequently, learning rules derived by differentiating global objective functions appear to lack biological plausibility. They may require non-local communication of information, or the computation of statistics averaged over an entire training pattern ensemble. This may be why the "top-down" approach arouses skepticism in many neurophysiologists. However, this approach may be viewed as a starting point for studying the computational processes involved in learning. Once the learning equations have been derived, they can often be refined into a more biologically plausible form, for example, using approximate statistics computed online.

1.1 Information Transmission Models

Is there any evidence that the brain optimizes a cost function? And if so, what form does that cost function take? Barlow [5, 6] hypothesized that early sensory processing serves to transform the highly redundant sensory signal into a more efficient *factorial code*; that is, the neurons' outputs are statistically independent when conditioned on the input. What cost function would cause a neural network to learn such a factorial code? A number of investigators (e.g. [4, 7, 8, 9]) have used concepts from information theory [10] to try to answer this question. The key idea in applying information theory to neural processing in these models is that the neuron can be viewed as a communication channel. It receives a signal along its input connections, and transmits an output signal. A cost function for learning is then set up in terms of the rate of information transmission through the channel, i.e., the mutual information (defined below) between the input and output signals, as shown in Figure 1 a). With all of these models, there are additional assumptions about the addition of noise at different points in the system. Atick and Redlich [7] proposed the following redundancy measure, which when minimized, approximates Barlow's optimality principle:

$$\mathrm{R} = 1 - \frac{I_{y;s}}{C(y)} \tag{1}$$

subject to the constraint of no information loss in the mapping from the input to the output, where $I_{y;s}$ is the mutual information between a unit's output y and the input signal s, and C is the channel capacity - the maximum achievable information that y may transmit about s. Atick [11] provides an excellent review of this and related approaches, as well as a good introduction to information theory. Here, we briefly review the definition of Shannon's mutual information measure [10], which will be used throughout this paper. The mutual information between two signals, y_A and y_B, is defined as follows:

$$\begin{aligned} I_{y_A;y_B} &= H(y_A) + H(y_B) - H(y_A, y_B) \\ &= -\int_{-\infty}^{\infty} p(y_A)\ \log\ p(y_A)\ dy_A\ -\ \int_{-\infty}^{\infty} p(y_B)\ \log\ p(y_B)\ dy_B \\ &\quad + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(y_A, y_B)\ \log\ p(y_A, y_B)\ dy_A\ dy_B \end{aligned} \tag{2}$$

where $H(x)$ stands for the entropy of the probability distribution $p(x)$ of a random variable x, and $H(x, y)$ stands for the entropy of the joint distribution $p(x, y)$ of random variables x and y. $H(x)$ is also referred to as the uncertainty in x. Shannon used this measure to characterize the amount of information flowing through a communication channel, and used the term *information rate* to refer to this quantity. To gain some insight into this measure, it is useful to consider the conditions under which it is maximal and minimal. A variable whose distribution has all its density concentrated in a very small region has very low entropy. It can be predicted with high certainty, and therefore conveys very little information. In contrast, the maximum entropy distribution is the uniform one[1]. A variable following the maximum entropy distribution is completely unpredictable, and conveys maximum information. The mutual information between two variables is highest when the variables have high entropies individually, but their joint distribution has low entropy. That is, each variable conveys a lot of information individually, but taken together they are redundant.

The class of approaches depicted in Figure 1 a) have been very successful at modelling the earliest stages of sensory processing, such as gain control in the blowfly compound eye [12], receptive field properties of the mammalian retina [7], and the emergence of centre-surround and oriented receptive fields in mammalian primary visual cortex [4]. Such cases lend support to the view of Barlow and others that the nervous system is optimized for transmitting information about the sensory input in an efficient, non-redundant manner.

[1]provided its density is confined to a finite volume [10]

Insert Figure 1 about here

Given the success of information-theoretic developmental models in accounting for early stages of sensory processing, a question naturally arises: To what extent can they model higher levels of processing? We now turn our discussion to a family of unsupervised learning algorithms called *Imax* [1] and related approaches. These learning algorithms use information-theoretic cost functions to discover *higher-order features* in sensory data. By higher-order features, we mean here features which could not be computed by a single-layer network, such as stereo disparity, or speaker-independent phoneme recognition. The remainder of the paper is organized as follows: In section 2, we explain how the Imax cost function encourages a network to discover higher-order features in multiple input streams. We then show how this algorithm can be realized in a number of ways, depending on the assumptions one makes about the probability distributions of the units' outputs. In each subsection, related learning algorithms in the literature are discussed. In section 3, we conclude with a general discussion of the utility of Imax in modelling the development of neural systems, in terms of computational complexity and biological plausibility, and we discuss several more biologically plausible related algorithms.

2 Imax: maximizing mutual information between outputs

The information available to our senses is highly redundant. Data compression, which reduces redundancy, appears to be one of the major achievements of the earliest stages of sensory processing. But can higher level perceptual abilities, such as visual object recognition and speech understanding, be modelled within an information-processing framework? Clearly, we do much more with the sensory information than just map it into an efficient code and store it away. The neural code must retain the information most useful to the organism, in a convenient form. At the same time, some of the sensory information, perhaps even most of it, is thrown away. This points to a counter-intuitive aspect of information theory as a basis for understanding neural representations: how is it that a completely random, maximally unpredictable signal can be the most informative one?[2] One problem with information theory is that it does not make any value judgements: that is, apart from discounting that which is redundant, it does not tell us what information is most useful, and what should be thrown away. Many other factors besides redundancy reduction may have a bearing on what and how information is represented in neural systems. Further,

[2]In fact, information theory is more consistent with our intuitions if we think of it in terms of the information *gained* by having observed a random signal at some point in time. After having observed a particular value of the signal, we gain much more information if the signal is white noise than if we had observed a more predictable signal.

different factors are likely to play a role in learning at different levels of the nervous system. At the level of sensory transduction, the primary objective is to reduce dimensionality while maximizing the information transmitted. At higher levels, factors such as prior knowledge about the structure in the world may play a part in how the nervous system filters information and shapes our perceptions. Prior knowledge could be incorporated in a number of ways, such as in the pattern of connectivity, or in the learning mechanisms. It is unlikely that very much *specific* knowledge is hard-wired into the cortex. On the other hand, neural systems may be predisposed to learn particular *kinds* of structure that are characteristic of sensory signals.

A ubiquitous feature of sensory data is coherence across time and across different sensory channels. By "coherence" we simply mean that one part of the signal can be somehow predicted from another part. The prediction may be based on an equality relation (e.g., the signal is equal for spatially or temporally adjacent samples), a linear relation, or any higher order relation. For example, in speech signals, individual words are typically composed of long intervals having relatively constant spectral characteristics corresponding to vowels, with short intervening bursts and rapid transitions corresponding to consonants. Even the consonants change across time in very regular ways. This temporal coherence at various scales makes speech predictable, to a certain degree. Similarly, visual, olfactory and tactile sensations exhibit coherence across space and time. In the visual domain, adjacent parts of the same object stimulate nearby photoreceptors in the retina. Nearby regions of the same object are usually coherent with respect to many parameters, such as texture, orientation, colour and depth; thus nearby photoreceptors tend to sample spatially coherent signals. Since most objects in the visual world move slowly, if at all, the visual scene changes slowly over time, exhibiting the same temporal coherence as other sensory sources. Further, there is coherence across different sensory modalities. When one examines an object, both visual and tactile cues can provide consistent information about features such as the object's texture, orientation, and even the identity of the object. When one listens to a speaker, the auditory signal may at times be unintelligible by itself, but visual cues such as the shape of the speaker's mouth can provide disambiguating information as to which word was spoken. Thus, it seems that spatio-temporal and multi-sensory coherence provide important cues for segmenting signals in space and time, and for object localization and identification.

In collaboration with Geoff Hinton, we derived a family of unsupervised learning algorithms called Imax [1, 13] that try to discover higher-order statistics in sensory data by taking advantage of spatio-temporal coherence. The central idea behind Imax is that two different neural units or neural network modules should learn to extract features that are coherent across their inputs, as shown in Figure 1 b). One way to get two modules to learn to agree is to minimize the squared error between their outputs. However, the modules could trivially minimize the error by always producing the same outputs for every input pattern. So we need an objective function that measures not only how well the outputs agree, but also whether they are detecting interesting features of their input. The cost function we arrived at is Shannon's mutual information measure.[3] However, we applied it in a way that is very different from the models depicted in Figure 1 a). Those models manipulate the mutual information between the inputs and outputs of a network, or in other words, the information transmission rate through a network. In contrast, the Imax algorithms maximize the mutual information between the outputs of two (or more) *different* network modules (see Figure 1 b). If there is any feature in common across the two sources, that feature should be discovered, while features which are independent across the two sources should be ignored.

From the definition in Equation 2, if two units' outputs have a high degree of mutual information, they must be individually unpredictable, but highly predictive of one another. From the latter, it can be inferred that the units are conveying information about a common source. In some situations, this is a very useful thing for neurons to do. However, in one extreme case, they might actually be driven by a common set of inputs. In this case, provided the shared inputs had high individual entropy, the goal of high mutual information could be achieved trivially by putting a large connection weight on one of their common input lines and all other weights equal to zero. Therefore, for this idea to work, the units must be connected to separate input channels. For example, they could receive input from spatially or temporally non-overlapping parts of the visual field, different frequency channels in the auditory system, or two different sensory modalities.

To see how the two approaches depicted in Figure 1 would achieve very different results on a concrete example, suppose the input is describable by four independent, equi-probable features: colour, texture, hardness, and shape. Further, suppose there are two input channels: a visual one sensitive to shape, texture and colour, and a tactile one sensitive to shape, texture and hardness. Thus the two input channels share information regarding only two properties, shape and texture. If each sensory pathway has two output units, then maximizing the information transmission rate separately for each pathway should result in each module's outputs representing a random subset of two out of three of its input features. Which particular input features each module learned to represent would depend on the initial weights. In contrast, if we maximized the mutual information between the two modules' outputs, they should both discover shape and texture, and ignore colour and hardness, because each of these properties is only received by one input channel. Although the latter features would contribute to raising the summed individual entropies of the two modules' outputs, they would lower the joint entropy by an equal amount, and therefore contribute nothing to the mutual information. This example illustrates how our mutual information cost function is much more constraining than maximizing the

[3]The choice of this mutual information objective function was inspired by the work of Peter Brown, Robert Mercer and colleagues on modelling the statistical regularities of text [14, 15]. They developed an unsupervised word classifier, based on the assumption that a word can be predicted more accurately from previous ones if the previous words can be grouped into equivalence classes based on their mutual information with the current word.

information transmission rate through the network, because it selects features which agree across multiple input channels.

One advantage of our mutual information measure for modelling learning in multiple input streams is that it provides a means for information in one channel to modulate learning in another channel, rather than simply merging the information in the two streams. Such a scheme might be useful, for example, to allow an auditory representation of spoken words to be kept in correspondence with a visual word form representation, and vice versa. Further, Shannon's information measure allows the comparison of two disparate information sources, which may have very different representations and even different dimensionality. This is in contrast to a measure such as the squared error between two representations, which is minimized only when the two codes are identical in both scale and form. Thus, sensory representations can be compared at a variety of levels in the system, and need not be first mapped into a common representation.

We now look at a number of ways in which our mutual information objective function can be instantiated in a family of learning algorithms called Imax. As can be seen in equation 2, the computation of the mutual information between two variables requires integration over their entire joint probability distributions. For continuous variables, in the general case, this is obviously not a feasible computation for neurons to perform. However, there are many special cases we can consider where the computation does become feasible. We will consider three of these cases in the remainder of this section, in which units' outputs are modelled as discrete binary variables, discrete multi-valued variables, and Gaussian variables respectively. Note that we do not need to make any assumptions about the inputs to the network; they could be either real-valued or binary.

2.1 Binary signals

One assumption often made in neural network models is that units are binary and probabilistic. Typically the sigmoid function (see Equation 3 below) is used to compute the probability of a unit's response, because it closely resembles the probability of a neuron generating an action potential as a function of membrane depolarization. We can easily estimate the mutual information between the outputs of two binary probabilistic units, by sampling their mean outputs and correlations over a large set of input cases. Once we have estimated the probabilities of each unit being on and the pair being on together, we can directly compute their mutual information. More importantly, we can then compute the derivative of their mutual information with respect to these probabilities, and with respect to the weights, to obtain weight update rules.

In the binary model considered here, each output unit computes a nonlinear probabilistic function of its real-valued total input $x_i = \sum_k w_{ik} y_k$, using the sigmoidal nonlinearity:

$$f(x_i) = \frac{1}{1 + e^{-x_i}} \tag{3}$$

For a particular input case α, the output of the ith unit in module A represents a binary variable y_{Ai}, which is on with probability $p^{\alpha}_{Ai} = f(x^{\alpha}_{Ai})$, and off with probability $p^{\alpha}_{\overline{Ai}} = 1 - f(x^{\alpha}_{Ai})$. However, rather than using stochastic output units and sampling each y_i many times per input pattern to estimate the p_is, we can simulate the learning much more efficiently by using the exact values of the p_is as the outputs.

The mutual information between two binary variables, y_{Ai} and y_{Bj}, can be computed as follows:

$$\begin{aligned} I_{y_{Ai};y_{Bj}} &= - p_{Ai} \log p_{Ai} - p_{\overline{Ai}} \log p_{\overline{Ai}} - p_{Bj} \log p_{Bj} - p_{\overline{Bj}} \log p_{\overline{Bj}} \\ &\quad + p_{Ai,Bj} \log p_{Ai,Bj} + p_{\overline{Ai},Bj} \log p_{\overline{Ai},Bj} + p_{Ai,\overline{Bj}} \log p_{Ai,\overline{Bj}} + p_{\overline{Ai},\overline{Bj}} \log p_{\overline{Ai},\overline{Bj}} \end{aligned} \tag{4}$$

Suppose we have two units in different modules, A and B - let's call them *neighboring units* for short - that want to maximize their mutual information. In order for them to do so, the above computation requires complete knowledge of the two terms in each unit's probability distribution, and the four terms in their joint distribution. However, each unit only needs to accumulate two ensemble-averaged statistics: the probability that it is on, and the probability that it is on together with its neighbor. The other terms in the distributions can be derived when the learning rule is applied. We can estimate the overall probability that the ith unit in module A is on by averaging its activity over the input sample distribution:

$$p_{Ai} = \langle y_{Ai} \rangle = \sum_{\alpha=1}^{N} P^{\alpha}\, p^{\alpha}_{Ai} \tag{5}$$

where N is the number of input samples and P^{α} is the prior probability of an input case α. For simplicity, from here on we will treat every case in the training set as equiprobable, so $P^{\alpha} = \frac{1}{N}, \forall \alpha$. Knowing p_{Ai}, we can now calculate the probability that the unit is off: $p_{\overline{Ai}} = 1 - p_{Ai}$. Similarly, the joint probability that two neighboring units are on simultaneously can be estimated by taking the expected value of the product of their states over the sample:

$$p_{Ai,Bj} = \langle y_{Ai}\, y_{Bj} \rangle = \sum_{\alpha=1}^{N} P^{\alpha}\, p^{\alpha}_{Ai}\, p^{\alpha}_{Bj} \tag{6}$$

From these quantities, the remaining three terms in the joint distribution can be computed: $p_{\overline{Ai},Bj} = p_{Bj} - p_{Ai,Bj}$, $p_{Ai,\overline{Bj}} = p_{Ai} - p_{Ai,Bj}$, and $p_{\overline{Ai},\overline{Bj}} = 1 - p_{Ai} - p_{Bj} + p_{Ai,Bj}$.

A weight update rule can be derived by differentiating $I_{y_{Ai};y_{Bj}}$ with respect to each incoming weight to each unit:

$$\frac{\partial I_{y_{Ai};y_{Bj}}}{\partial w_{ik}} = - \sum_{\alpha} P^{\alpha} \left[\log \frac{p_{Ai}}{p_{\overline{Ai}}} - p^{\alpha}_{Bj} \log \frac{p_{Ai,Bj}}{p_{\overline{Ai},Bj}} - p^{\alpha}_{\overline{Bj}} \log \frac{p_{Ai\overline{Bj}}}{p_{\overline{Ai},\overline{Bj}}} \right] p^{\alpha}_{Ai}\, p^{\alpha}_{\overline{Ai}}\, y^{\alpha}_{k} \tag{7}$$

Full details of this derivation are given in [16].

How might such a learning rule be instantiated in biological machinery? As a first approximation, the statistic p_{Ai} could be estimated incrementally, as a decaying average of a unit's probability of responding. This statistic requires only local information. On the other hand, the computation of $p_{Ai,Bj}$ is non-local in the sense that it requires the ith unit to observe the jth unit's state (and vice versa), without actually being driven by a synaptic connection from the jth unit. This might be accomplished, for example, with non-driving connections that modulate plasticity. A simple Hebb rule applied to such a link would be sufficient to compute $p_{Ai,Bj}$. Phillips, Kay and Smyth [17] have proposed a more biologically plausible version of binary Imax they call *coherent Infomax*, in which the outputs from one processing stream modulate the activity in another stream, while the mutual information between the two streams is maximized. Kay [18] extends this analysis to modules having multiple outputs, and presents a number of locally computable approximations for this case. Another way to avoid the problem of communication across non-driving links is to maximize the mutual information between a single module's outputs at different points in time. This idea makes more sense when applied to discrete, multi-valued distributions rather than binary distributions, and will be discussed further in the next subsection.

We tested the Imax algorithm for binary units on a problem known to be impossible for neural networks without hidden layers: the binary shift problem (details of these simulations are reported in [16]). The main point of this demonstration was to show how Imax could learn to solve a problem that is not linearly separable.[4] The problem can be described as follows: given two strings of bits, the second one being a shifted version of the first, what is the shift? This problem may sound very artificial, but in fact it captures the essence of the stereo disparity detection problem: given two different views of the same image, what is the disparity between them at each region of the image?

We can apply Imax to this problem by dividing the input into local receptive fields, as shown in Figure 2. Each network module receives input from a different receptive field. Corresponding neighbors Ai and Bi try to maximize their mutual information. We found that when the first layer of the network shown in Figure 2 was trained in this manner, units tended to become somewhat tuned to shift. That is, they tended to detect particular shifts at particular locations. Because units were only trained to maximize pairwise mutual information, there was nothing to prevent redundancy in the first layer. But because of differences in their initial random weights, units tended to develop a variety of receptive field profiles. When a second layer of units was trained in exactly the same manner, these units became even better at discriminating left from right shifts independently of position. Although the small network shown in Figure 2 only learned to perfectly discriminate left

[4]Note, Imax is not "learning to solve" any specific problem in the sense that a supervised learning procedure would do explicitly. It is just learning to maximize mutual information, and in so doing, it may end up implicitly discovering a solution to a difficult nonlinear problem.

from right shifts after 300 learning iterations on 27 of 50 runs, second layer units always became highly shift-tuned. Adding more units in the first layer generally caused the second layer to do better. Presumably a sufficiently large network would nearly always solve this problem perfectly.

Insert Figure 2 about here

When the units in the first layer were further trained by back-propagating the mutual information derivatives from the top layer, the network shown in Figure 2 learned to discriminate shift perfectly after 300 iterations on 48 of 50 runs. Another way to propagate knowledge from the top layer down to lower layers, rather than back-propagating derivatives, would be to have the lower layer units maximize mutual information with units in the upper layers as well as with units in the same layer. To avoid local maxima, this would have to be an approximate, asymmetric maximization of mutual information, i.e. the upper layer units would ignore their mutual information with the lower layer units. As the upper layer units began to catch on to the global shift across the receptive field, they could provide more discriminating information to the lower layer units. A similar scheme is employed in de Sa and Ballard's Minimizing Disagreement algorithm [19], described further in the discussion section.

The binary shift example illustrates how Imax can be used to detect nonlinear binary features such as stereo disparity in visual images. Phillips et al [17] demonstrated that it can also discover that edge contrast is coherent across two input streams. In the next subsection, we show how to generalize the learning procedure to model multi-valued features.

2.2 Discrete multi-valued signals

Binary Imax is limited to learning discrete two-valued features like left versus right shift. However, features in real sensory data are more often continuous than binary. One way for a group of units to represent a continuous feature is by a population code, where each unit responds to a limited range of values of the feature. For example, orientation is thought to be represented this way in primary visual cortex; each orientation-tuned neuron has a preferred range of orientations that it responds best to. This kind of representation can be modelled as a discrete random variable y_A that takes on one of m possible values. A population code maps onto this model nicely if we interpret the units' outputs in each module as representing a probability distribution over the possible feature values. Each unit's expected output for pattern α, p^{α}_{Ai}, represents the probability that the variable takes on one of the m possible values (or falls within one of m sub-ranges of values). For this probabilistic interpretation of the units' outputs, we must choose an activation function that forces their activities to be positive and sum to one. One possible choice is the "softmax" activation function [20]:

$$p^\alpha_{Ai} = \frac{e^{x_i}}{\sum_{j=1}^m e^{x_j}} \tag{8}$$

where x_i is the total weighted summed input to the ith unit.

The mutual information between two m-valued variables, y_A and y_B, can be computed as follows:

$$I_{y_A;y_B} \;=\; -\sum_i p_{Ai} \log p_{Ai} \;-\; \sum_j p_{Bj} \log p_{Bj} \;+\; \sum_{ij} p_{Ai,Bj} \log p_{Ai,Bj} \tag{9}$$

By differentiating $I_{y_A;y_B}$ with respect to each weight w_{il}, we obtain the following weight update rule, shown here for the lth weight to the ith unit in module A:

$$\frac{\partial I_{y_A;y_B}}{\partial w_{il}} \;=\; -\sum_\alpha P^\alpha \; p^\alpha_{Ai} \; \sum_j p^\alpha_{Aj} \left[\log \frac{p_{Ai}}{p_{Aj}} \;-\; \sum_k p^\alpha_{Bk} \; \log \frac{p_{Ai,Bk}}{p_{Aj,Bk}} \right] \; y^\alpha_l \tag{10}$$

As in the binary case, for each unit, we need to know the probability that it is on, p_{Ai}. We also need to know the joint probability that it is on together with each of its neighbors: $p_{Ai,Bj}$. This time, however, we have two groups of m units rather than just one pair of units maximizing I. So the joint probability distribution now consists of a matrix of m^2 terms rather than just four terms. This complexity can be greatly reduced by a simple approximation in computing the joint entropy [21]: instead of considering all m^2 terms in the joint probability matrix, we assume that only the diagonal terms $P_{Ai,Bi}$ are significant, and make the maximum entropy assumption about the remainder of the distribution. This amounts to deciding a priori that the ith units in each module should have a high probability of being on together, for i ranging from 1 to m, and we don't care how correlated the ith and jth units are when $i \neq j$. Further, each unit can make its own local approximation to the mutual information, making the maximum entropy assumption about all the other units in the network except its neighbor, subject to the constraints that $\sum_j P_{Ai,Bj} = P_{Ai}$ and $\sum_i P_{Ai,Bj} = P_{Bj}$. Then, as in the binary case, each unit only needs to keep track of its own probability of being on, as well as the probability that it is on concurrently with its neighbor. The mutual information, from the perspective of the ith pair of neighbors, can now be computed as follows:

$$\begin{aligned} I_{y_A;y_B} \approx \; & -p_{Ai} \log p_{Ai} \;-\; (1-p_{Ai}) \log \frac{1-p_{Ai}}{m-1} \;-\; p_{Bi} \log p_{Bi} \;-\; (1-p_{Bi}) \log \frac{1-p_{Bi}}{m-1} \\ & + p_{Ai,Bi} \log p_{Ai,Bi} \;+\; p_{\overline{Ai},Bi} \log \frac{p_{\overline{Ai},Bi}}{m-1} \;+\; p_{Ai,\overline{Bi}} \log \frac{p_{Ai,\overline{Bi}}}{m-1} + p_{\overline{Ai},\overline{Bi}} \log \frac{p_{\overline{Ai},\overline{Bi}}}{(m-1)^2} \end{aligned} \tag{11}$$

The above is nearly identical to the binary case (Equation 5), except for the weighting of some of the probabilities by $\frac{1}{m-1}$, which enforces the constraints mentioned above. To get a consistent global objective function for learning, we can maximize the sum of the mutual information terms computed locally by each unit within a module.

The weight update rule is more complicated now than it was for a pair of binary units because of the softmax activation function (Equation 8), which makes the activity of each unit in a module depend on all the other units in same layer of the module. However, we can break down the computation of the weight updates into two steps, so that units within a module only need to communicate one piece of information with each other:

$$\begin{aligned}\frac{\partial I_{y_A;y_B}}{\partial p^\alpha_{Ai}} &= P^\alpha \left(\log \frac{(1-p_{Ai})}{(m-1)p_{Ai}} \;+\; p^\alpha_{Bi} \log \frac{(m-1)^2 \, p_{Ai,Bi} \;\; p_{Ai,\overline{Bi}}}{p_{\overline{Ai},Bi} \;\; p_{\overline{Ai},\overline{Bi}}} \right) \\ \frac{\partial I_{y_A;y_B}}{\partial w_{ij}} &= \sum_\alpha \left(\frac{\partial I_{y_A;y_B}}{\partial p^\alpha_{Ai}} - \sum_k p^\alpha_{Ak} \frac{\partial I_{y_A;y_B}}{\partial p^\alpha_{Ak}} \right) p^\alpha_{Ai} \;\; y^\alpha_j \end{aligned} \tag{12}$$

Alternatively, each unit i could ignore the terms $\frac{\partial I_{y_A;y_B}}{\partial p^\alpha_{Aj}}$ for $j \neq i$, and we would have a learning rule very much like the binary case, though it would only be following the derivative to Equation 11 in an approximate sense.

2.3 New simulation results for discrete Imax

The learning procedures defined by equations 10 and 12 have been previously applied to several problems [13, 21]. We will now describe some new results that illustrate the strengths and weaknesses of discrete Imax. The first set of results shows how the learning procedure can discover non-linear features from temporally coherent visual stimuli, and the second shows how it performs on a real speech classification problem.

One way to get Imax to learn that stimuli are temporally coherent is to maximize the mutual information between the outputs of a single network module at successive time steps, $I_{y(t);y(t-1)}$. Each unit maintains a short-term activity trace, allowing it to compare its recent activation level with that resulting from the current pattern. Any of the learning procedures discussed in this section would then be applicable. In the simulations reported here, we applied the learning rule given by equation 12, with the first line of that equation modified as follows to apply to the time domain:

$$\frac{\partial I_{y(t);y(t-1)}}{\partial p^t_i} = P^t \left(2 \, \log \frac{(1-p_i)}{(m-1)p_i} \;+\; (p^{t-1}_i + p^{t+1}_i) \log \frac{(m-1)^2 \, p_{i,i} \;\; p_{i,\bar{i}}}{p_{\bar{i},i} \;\; p_{\bar{i},\bar{i}}} \right) \tag{13}$$

where $p_{i,i}$ now denotes $P(y_i(t) = 1, y_i(t-1) = 1)$ etc.

To demonstrate the capabilities of this learning procedure, we devised a simple temporal signal classification problem. We used a set of training patterns of drifting sinusoidal gratings as shown in Figure 3. Networks with a single module with two layers of units were trained using the learning rule given by equation 13. Units in each layer used the softmax activation function (Equation 8).

Insert Figure 3 about here

The first layer of units was trained first. Not surprisingly, these units learned to detect particular spatial frequency and phase combinations. Figure 4 shows the tuning curves of the first nine of these units on a typical run. In this case, the network had a total of twenty units in the fist layer, and amongst them, they spanned the entire phase-frequency space represented in the training set. This is not too surprising because the patterns were moving very slowly across the "visual field" of the network, so a unit with a relatively narrow tuning curve in any part of the phase-frequency space would tend to respond consistently over time. More interesting was what happened when a second layer of units was then trained. There were three frequencies present in the training data, so we used only three units in the output layer, to see whether they would learn to classify the inputs by frequency, independent of phase. Figure 5 shows activity histograms for the three second layer units plotted against frequency for a typical run. The histograms show that the second layer units are very sharply tuned to particular frequencies, but their responses are phase-invariant.

Insert Figure 4 about here

Insert Figure 5 about here

The fact that Imax can learn to perform shift-invariant pattern classification is significant because this is something a single layer network could not do; the problem is not linearly separable. It is somewhat surprising that the second layer units performed as well as they did. While the temporal coherence in the training set spanned a fairly long time scale (50 patterns), the patterns varied very little at short time scales, and the network was only trying to maximize mutual information across a very short time scale of two time steps. Nonetheless, the network was able to come close to the globally optimal solution of separating the patterns according to spatial frequency. This suggests that Imax could learn to recognize more complex patterns in the same manner. However, we cannot claim that the network learned a general solution to the viewpoint-invariant object recognition problem. The first layer units learned to recognize specific "views" of each pattern class, and then the second layer merely learned to group these views together to achieve translation-invariant classification. There is much debate in the vision literature as to whether this constitutes a plausible account of more complex forms of object recognition (see e.g. [22]).

The second set of results we will describe is for a speaker-independent vowel recognition task. We used the Peterson-Barney data set consisting of the first two formants extracted from ten vowels spoken by multiple male and female speakers [23]. These data are shown in Figure 6. Although some of the vowel classes are linearly separable, there is much overlap between many of the classes. We modified the vowel recognition task to simulate multi-modal learning, as in [24], by presenting two network modules simultaneously with examples of spoken vowels. On each case, they received as input a pair of examples randomly drawn from the same vowel class. This problem is meant to resemble the one people face in trying to recognize speech from a combination of auditory and visual inputs. Usually, the visual and auditory inputs are in close correspondence. Often, one of the two channels is ambiguous and the other can be used to disambiguate what was spoken. However, in our training set, the problem is made even more difficult by the fact that because the classes overlap, the paired auditory inputs can be both ambiguous.

Insert Figure 6 about here

In initial simulations using the approximate Imax learning rule given by equation 12, the network tended to converge rapidly to sub-optimal solutions in which several "greedy" units' weights grew very large, while the other units weren't able to capture any patterns. The greedy units each learned to capture two or more classes of vowels. The network thus discovered that it could do a good job of minimizing the uncertainty between the two modules' outputs by merging many vowel classes together, at the expense of the individual output entropies of the modules. This highlights a drawback of the approximate mutual information (Equation 11): it tries to compress all of the density in the joint probability matrix for the two modules into the diagonal terms, $p_{Ai,Bi}$. If two units i and j move their weights toward different classes that overlap, there will be non-zero off-diagonal terms in the joint density, $p_{Ai,Bj}$, that tend to lower the approximate mutual information, even though this would tend increase the true mutual information. We therefore maximized the exact discrete mutual information (Equation 9). However, there was still a tendency for some of the weights to grow very large and the learning to grind to a halt. We found that a better choice of activation function prevented this from happening. Instead of using the weighted summed input for x_i in the softmax activation (Equation 8), we used an appropriately normalized Gaussian activation function, which is relatively insensitive to the scale of the weights:

$$p_{Ai}^{\alpha} = \frac{e^{x_i^{\alpha}}}{\sum_{j=1}^{m} e^{x_j^{\alpha}}} \tag{14}$$

$$x_i^{\alpha} = \frac{\| \vec{y}^{\alpha} - \vec{w}_i \|}{\sigma_i^2} \tag{15}$$

where $\| \cdot \|$ is the L2 norm, $\vec{w}_i$ is the ith weight vector, $\vec{y}^{\alpha}$ is the input vector on case α, and the variance of each unit, σ_i^2, is an online approximation to the true variance of the input about the unit's weight vector: $\sigma_i^{2^{\alpha}} = k \sum_j (w_{ij}^2 + y_j^2)$. This approximation to the variance would be exact (to within a constant factor k) if the input vectors were of fixed length and uncorrelated with the weight vectors. In the simulations reported here, we used $k = 0.1$

The learning rule for maximizing the discrete mutual information (Equation 9) using Gaussian activation functions is:

$$\frac{\partial I_{y_A;y_B}}{\partial w_{il}} = -\sum_{\alpha} P^{\alpha}\, p^{\alpha}_{Ai} \sum_{j} p^{\alpha}_{Aj} \left[\log \frac{p_{Ai}}{p_{Aj}} - \sum_{k} p^{\alpha}_{Bk}\, \log \frac{p_{Ai,Bk}}{p_{Aj,Bk}}\right] (y^{\alpha}_{l} - w_{il} + w_{il}\, x^{\alpha}_{i}) \tag{16}$$

Using this learning rule, Imax networks with varying numbers of units were trained for 5000 iterations with a learning rate of 0.1 and momentum of 0.9 on the Peterson-Barney vowels. The representations learned by Imax were compared against those obtained by fitting a mixture of Gaussians model using the EM algorithm [25] for 1000 iterations. This latter model, or the equivalent (e.g., Soft Competitive Learning [26]), is often used to preprocess complex signal data for classification tasks such as speech recognition. We therefore preprocessed the patterns by the two methods, and then fed their outputs to single layer supervised nonlinear back-propagation networks. The performance of the classification networks are summarized in Table 2.3. For networks with five or more Gaussians, Imax always outperformed EM. EM typically converged to a solution where three or four of the Gaussians captured most of the data and had fairly large variances, while the other Gaussians captured the remaining "outliers" and had much smaller variances. While this is a good description of the overall distribution of the data (see Figure 6), it does not capture the ten individual vowel classes particularly well. On the other hand, Imax tended to place the Gaussian centres in the middle of individual vowel classes or between pairs of vowel classes, and therefore led to better discrimination. A second version of EM, with all the variances constrained to be equal (EM1), was then tested. These results came much closer to, or in one case equalled the performance of Imax networks of 5 units or larger, as shown in Table 2.3. Nowlan [26] has reported a performance level of 82.6% on these data using a Soft Competitive Learning (SCL) network of 20 spherical Gaussian units with a single variance.

For the network with only three Gaussians, however, Imax performed much worse than EM and EM1. It is interesting to see exactly how each algorithm divided up the vowel classes in this case. The Gaussian centres learned by EM1 and Imax for this case are shown in Figure 6. Notice that the data span a region of the input space shaped roughly like a rightward pointing arrowhead. The EM objective function drives the network to learn a good model of the overall shape of this distribution. As the figure shows, it does so fairly well in this case by placing one Gaussian in the middle of the pointed tip of the arrowhead, and the remaining two Gaussians in the middle of the two tails of the arrowhead. These three Gaussian centres form a good basis for interpolating the location of any point in the distribution. Imax, on the other hand, tries to position each of its Gaussians by merging several vowel classes together so as to form three roughly equal-sized, non-overlapping "super-classes", as shown in Table 2. It is then much more difficult for a supervised

Method	Percent Correct			
	3 rbfs	5 rbfs	10 rbfs	20 rbfs
Imax	56.36	71.55	76.37	78.45
EM	67.40	69.54	75.71	77.44
EM1	70.21	71.28	76.37	77.91

Table 1: Training set performance of single-layer supervised networks in classifying the Peterson-Barney vowel data. The data was preprocessed by unsupervised networks with Gaussian units, commonly known as radial basis functions (RBFs) trained by three different methods: 1) Imax, 2) EM with spherical Gaussians (each Gaussian adapts its variance), and 3) EM1 (1 variance for all Gaussians: each Gaussian adapts its variance, and then they are averaged). To generate more distributed (less binary) input vectors for the supervised networks, the variances of the Imax units were increased by a factor of 5, and those of the EM Gaussians were increased by a factor of 10 before they were used to preprocess the data; this substantially improved the classification performance in both cases. The supervised networks were trained by the method of conjugate gradients with line search for a maximum of 5000 iterations (fewer if the line search converged sooner), to minimize the asymmetric divergence between the network outputs and target classification vectors. The criterion for correct performance was that the unit representing the correct class should be the most active of all output units.

classifier trained on the output of the Imax network to discriminate between those classes which have been merged. This illustrates a point that was made in the introduction about unsupervised learning procedures for low-level feature extraction versus higher-order feature extraction and classification. For early feature extraction, it is best to have a very general representation that captures as much information as possible in the input. This is especially true if it is not known in advance exactly what tasks the representation will be used for. Density estimation procedures like EM and SCL are good at this. On the other hand, to achieve good high-level object or speech recognition, we need some way to map low-level features into a representation that is invariant with respect to location and scale. This necessarily entails a loss of information, but results in a more powerful representation for a solving a specific class of tasks, namely, object identification.

A compromise between Imax and a mixture of Gaussians model is possible. Ghahramani [27] developed such a model for the aligned formation of multiple topographic maps in different sensory streams. The cost function combined four energy terms: (i) a term that measures the model fit for each unit; (ii) a term that constrains nearest neighbors within a modality to have similar weights; (iii) a term that encourages a one-to-one correspondence between activities in the two maps; and (iv) a term that incorporates the discrete mutual

Unit 1	Unit 2	Unit 3
0.00	1.00	0.00
0.00	1.00	0.00
0.02	0.97	0.01
0.83	0.16	0.01
0.96	0.00	0.04
1.00	0.00	0.00
0.81	0.00	0.19
0.11	0.00	0.89
0.03	0.01	0.96
0.09	0.25	0.66

Table 2: Average activity of each of the three Gaussian units for each of the ten vowel classes. The network was trained to maximize the discrete mutual information between the three outputs of two neighboring modules.

information between the outputs of the two modalities. Ghahramani's final cost function was the log likelihood of the data, computing the model probability as a function of the total energy (a sum of the four constraints) under the Boltzmann distribution. The model was fitted using the EM algorithm. Ghahramani showed that this learning procedure could develop aligned topographic maps when applied to two input streams representing polar and Cartesian two-dimensional co-ordinates.

To summarize the findings reported in this subsection, discrete Imax was able to solve a problem that is not linearly separable (phase-invariant spatial frequency detection) by maximizing mutual information over time in a two-layer network. On a real signal analysis problem (vowel discrimination) Imax outperformed a mixture of Gaussians model except for very small networks. The mixture model develops a more general representation which is good for minimizing the reconstruction error of the data. From this representation, the supervised network is then better able to extract the underlying classes in the data. For very small networks, this latter solution is preferable. As a first step in preprocessing complex signals, it is better to remain uncommitted about where to draw classification boundaries. On the other hand, Imax may be a better model of higher-order feature extraction. It learns classification boundaries that are more closely tied to structure that is coherent across different input channels, and is likely to reflect important cues for perception.

2.4 Imax For Continuous Outputs

The discrete Imax algorithms described above are good for modelling classification problems, where the output of a neuron represents a binary feature. However, features like stereo disparity and spatial frequency, which are important for building a representation of the visual world, vary continuously. Thus, a continuous representation may be more natural in signal processing domains. To get a tractable expression for the mutual information between two continuous signals, we can make various restrictive assumptions about the network, or the units' output distributions. One particular model which I have explored in depth, in collaboration with Geoff Hinton [1, 16], makes the assumption that the units' activities have Gaussian distributions. We further assume that there is a common underlying Gaussian signal embedded in each module's input, but the two channels are corrupted by independent, Gaussian noise. The mutual information between a pure Gaussian signal, s, and a random variable equal to the signal corrupted by additive Gaussian noise, $y = s + n$, is just the log of the signal to noise ratio [10]:

$$I_{y;s} \quad = \quad 0.5 \log \frac{V(s+n)}{V(n)} \tag{17}$$

where $V()$ denotes variance. In our case, we consider the shared information between two neurons' outputs, each corrupted by different noise terms: $y_a = s + n_a$, $y_b = s + n_b$. The mutual information between the two signals is:

$$I_{y_a;y_b} = 0.5 \log \frac{V(y_a)V(y_b)}{V(y_a)V(y_b) - V(\frac{y_a+y_b}{2}) + V(\frac{y_a-y_b}{2})} \tag{18}$$

The weight update rule for a weight to a unit in module A is as follows:

$$\frac{\partial I_{y_a;y_b}}{\partial w_{ij}} = \sum_{\alpha=1}^{N} \frac{1}{N} \left[\frac{y_a^\alpha + y_b^\alpha - \langle y_a + y_b \rangle}{V(y_a + y_b)} - \frac{(y_b^\alpha - y_b^\alpha) - \langle y_a - y_b \rangle}{V(y_a - y_b)} \right] y_a^\alpha (1 - y_a^\alpha) x_j \tag{19}$$

where x_j is the jth input to the unit whose output is y_a, and $\langle\rangle$ denotes the expectation. See [13] for details of the derivation of the above information measure and learning rule, and some alternative mutual-information based cost functions for this model.

2.5 Relation to canonical correlation

A standard statistical method called canonical correlation (see, e.g. [28]) can be shown to be closely related to the maximization of Equation 18 [29, 13]. Given two sets of input variables, $\mathbf{x_a}$, and $\mathbf{x_b}$, canonical correlation analysis finds linear combinations $y_a = \mathbf{w_a^T x_a}$ and $y_b = \mathbf{w_b^T x_b}$ that will maximize the correlation between y_a and y_b:

$$\rho(y_a, y_b) \quad = \quad \frac{\langle (y_a - \overline{y_a})(y_b - \overline{y_b}) \rangle}{\sqrt{\langle (y_a - \overline{y_a})^2 \rangle \, \langle (y_b - \overline{y_b})^2 \rangle}} \tag{20}$$

It is easy to show that for linear networks, maximizing Equation 18 is equivalent to performing canonical correlation analysis. Since ρ does not depend on the scale of y_a and y_b, maximizing ρ is equivalent to maximizing $\langle (y_a - \overline{y_a})(y_b - \overline{y_b}) \rangle$ subject to $V(y_a) = V(y_b) = 1$. Further, since ρ is invariant with respect to translations of its arguments, we can constrain the variables y_a and y_b to have zero means as well as unit variances, and simply maximize $\langle y_a y_b \rangle$. Equation 18 is also invariant with respect to scaling and translations of the variables y_a and y_b. If we again constrain the variables to have means of zero and unit variances, we find that $V(y_a + y_b) = 2\,(1 + \langle y_a y_b \rangle)$ and $V(y_a - y_b) = 2\,(1 - \langle y_a y_b \rangle)$. From this, it is easy to show that maximizing mutual information for Gaussian variables (Equation 18) is equivalent to maximizing $\langle y_a y_b \rangle$ subject to $V(y_a) = V(y_b) = 1$ and $\overline{y_a} = \overline{y_b} = 0$, which is equivalent to maximizing ρ.

The difference between continuous Imax and canonical correlation is that canonical correlation is a linear procedure that can be solved analytically. On the other hand, Imax maximizes an equivalent cost function in a nonlinear network which may have multiple layers. To show how they arrive at very different solutions in general, we applied both methods to the binary shift problem described in section 2.1. Canonical correlation reached a degenerate solution in which all the weights were zero. This is because there is no possible linear transformation of the binary input vectors that would make them more correlated. On the other hand, Imax produced outputs which are perfectly correlated with the shift, as shown in Table 3.

2.6 Applications of continuous Imax

Unlike the binary case, continuous Imax can learn to extract real-valued non-linear features. We created a continuous version of the shift problem by creating random dot stereograms of curved surfaces with continuously varying disparities [1, 16]. The network was able to form a representation of the disparities in the images by learning local position-specific features at particular disparities in the first layer, and combining these in the second layer to form a position-invariant representation of continuous disparity. A typical disparity tuning curve for an output neuron is shown in Figure 7. We also showed how continuous Imax can learn to interpolate disparity across multiple receptive fields, in images of curved surfaces.

Insert Figure 7 about here

One way to extend the model to modules with multiple outputs is to assume a Gaussian mixture model of the underlying signal; this allows a group of units to form a population code for a continuous real-valued feature of their inputs. In joint work with Geoff Hinton [30], when we applied this idea to random dot stereograms, each output unit learned to become tuned to a different range of disparities.

Run	$\rho(y_a, shift)$
1	-0.999921
2	0.993861
3	0.999608
4	-0.998067
5	0.998734
6	0.998166
7	-0.99953
8	-0.998192
9	0.999416
10	0.998595

Table 3: The correlation between one of the outputs of a multi-layer network and the shift after learning, when trained for 10 runs with continuous Imax on the binary shift problem. The network consisted of two modules, each with 10 nonlinear hidden units and one linear output unit. On each run, the first layer of the network was first trained for ten iterations; corresponding pairs of units in the two modules maximized Equation 15. The second layer was then trained to maximize the same objective function for ten iterations using the method of conjugate gradients to speed up the learning, with back-propagation of mutual information gradients to further train the first layer units.

Several other investigators have applied variants of Imax to other problems. For example, Stone [31] proposed a cost function that maximizes the log ratio of the long-range output variance of a unit to the short-range variance. This is an approximation to the mutual information between the output of a unit on the current time step and the unit's output averaged over a recent time interval [31]; thus it is an approximation to Imax in the time domain, as described in the previous section for discrete Imax. Stone showed that the learning rule that maximizes his cost function is a Hebb-like rule that has a simple, online approximation. Stone applied this learning procedure to a temporal version of the stereo problem, and showed that his network could extract disparity from real stereo images.

Ukrainec and Haykin [32, 33] successfully applied an algorithm closely related to Imax to a difficult signal processing problem for navigation. They used a radar system called Polarimetric Radar for Accurate Navigation (PRAN) invented by Haykin [34], in which a set of polarization-twisting reflectors, situated at known points along a confined waterway, allow a ship to determine its location. A pair of orthogonally polarized antennas on board the ship picks up the radar returns from the reflectors. The important characteristic of this system for neural processing is that because each reflector (target) is polarized, the two detectors will differentially respond to the target signal. The problem for the neural network is to discriminate features that *differ* in the two input channels rather than a common feature. Ukrainec and Haykin therefore *minimized* the mutual information between the outputs of two modules, subject to a constraint that prevented the weights from collapsing to zero. The network was thereby able to learn to detect the component of the input signal that was independent across the two channels: the target. After preprocessing the input with a mixture of Gaussians model (RBF network) using the EM algorithm, the mutual information minimizing algorithm was applied. This system outperformed by a factor of two a linear Principal Component Analyzing (PCA) network, which in turn nearly doubled the performance of a conventional cell-averaging constant false alarm rate (CA-CFAR) processor commonly used in radar systems [32]. In further work [33], it was found that a hybrid system combining the RBF-mutual information network with an adaptive cancellation system, followed by a post-processing CFAR stage, outperformed all of the above methods individually.

2.7 Imax between multivariate continuous outputs

The continuous Imax algorithm described above applies to networks which extract a single feature that is predictive across their two input channels. When this learning algorithm is applied simultaneously to multiple pairs of units, there is nothing to prevent all the units in each module from doing exactly the same thing. Of course, differences in random initial weights may result in different features being extracted. However, if there are multiple features that vary in complexity, the lowest-order features are likely to be discovered to the exclusion of higher-order features. For example, the average intensity in visual images is a trivial spatially coherent feature that can be computed by a single unit, whereas a feature like stereo disparity requires a network with nonlinear hidden units.

One way to train networks with multiple output units is to compute the mutual information between two vectors of variables. If the outputs are assumed to represent a multivariate Gaussian with each element corrupted by independent Gaussian noise, $\mathbf{y_a} = \mathbf{s} + \mathbf{n_a}$, $\mathbf{y_b} = \mathbf{s} + \mathbf{n_b}$, we can easily extend equation 18 to the multivariate case. Zemel and Hinton [35] applied this idea to an object recognition problem. The cost function they maximized was the mutual information between the mean of the output vectors and the common underlying signal:

$$I_{\mathbf{y_a}+\mathbf{y_b};\mathbf{s}} = \log \frac{|Q_{y_a+y_b}|}{|Q_{y_a-y_b}|} \tag{21}$$

where $\mid Q_x \mid$ denotes the determinant of the covariance matrix of x. The weight update rule is much more complicated in this case, because it involves the computation of determinants. However, it is consequently much more powerful, because it can discover multiple uncorrelated features. In Zemel and Hinton's simulations, the two modules received as input neighboring regions of images of simple randomly oriented two-dimensional objects.

Their network was able to learn a multivariate representation of the viewing parameters of the objects (i.e., the rotation, scale, and translation in two dimensions).

A more powerful generalization of Imax would be to extract multiple *independent* features rather than just uncorrelated features. Bell [36] showed how to compute the output entropy of a network with multiple sigmoid units. For a network that computes an invertible, monotonic function of its inputs, the output entropy can be computed directly as follows:

$$H(\mathbf{y}) \;=\; \langle \log abs(|J|) \rangle \;+\; H(\mathbf{x}) \tag{22}$$

where $abs()$ denotes the absolute value, and $|J|$ is the Jacobian of the transformation computed by the network (the determinant of the matrix of partial derivatives: $J_{ij} = \frac{\partial y_i}{\partial x_j}$). In order for the Jacobian to be non-zero, the output of the network must be of the same dimensionality as the input.

We can apply Bell's analysis to Imax, but unfortunately, it leads to a rather negative result. For two modules of m units, each with m inputs $\mathbf{x_a}$ and $\mathbf{x_b}$, and m outputs, $\mathbf{y_a}$ and $\mathbf{y_b}$, Imax is equivalent to maximizing the information shared between the inputs, I_{x_a,x_b}. If the Jacobians for each module are $|J_a|$ and $|J_b|$, and the Jacobian for the entire network is $|J_{ab}|$, the mutual information between the two sets of outputs is:

$$I_{\mathbf{y_a};\mathbf{y_b}} \;=\; H(\mathbf{y_a}) + H(\mathbf{y_b}) - H(\mathbf{y_a},\mathbf{y_b}) \tag{23}$$

The individual entropies, $H(\mathbf{y_a})$ and $H(\mathbf{y_b})$, can be computed using equation 22:

$$\begin{aligned} H(\mathbf{y_a}) &= \langle \log abs(|J_a|) \rangle \;+\; H(\mathbf{x_a}) \\ H(\mathbf{y_b}) &= \langle \log abs(|J_b|) \rangle \;+\; H(\mathbf{x_b}) \end{aligned} \tag{24}$$

The joint entropy can be computed as follows:

$$\begin{aligned} H(\mathbf{y_a},\mathbf{y_b}) &= \langle \log abs\,(|J_{ab}|) \rangle \;+\; H(\mathbf{x_a},\mathbf{x_b}) \\ &= \langle \log abs \left(\begin{vmatrix} \mathbf{J_a} & \mathbf{0} \\ \mathbf{0} & \mathbf{J_b} \end{vmatrix} \right) \rangle \;+\; H(\mathbf{x_a},\mathbf{x_b}) \\ &= \langle \log abs \left(\begin{vmatrix} \mathbf{J_a} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{vmatrix} \; \begin{vmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{J_b} \end{vmatrix} \right) \rangle \;+\; H(\mathbf{x_a},\mathbf{x_b}) \\ &= \langle \log abs\,(|\mathbf{J_a}|\;\;|\mathbf{J_b}|) \rangle \;+\; H(\mathbf{x_a},\mathbf{x_b}) \\ &= \langle \log abs\,(|\mathbf{J_a}|) \rangle \;+\; \langle \log abs\,(|\mathbf{J_b}|) \rangle \;+\; H(\mathbf{x_a},\mathbf{x_b}) \end{aligned} \tag{25}$$

where $\mathbf{1}$ is the $m \times m$ dimensional identity matrix, and $\mathbf{0}$ is the $m \times m$ dimensional matrix of 0's. Substituting 24 and 25 into 23, we have:

$$\begin{aligned} I_{\mathbf{y_a};\mathbf{y_b}} &= H(\mathbf{x_a}) + H(\mathbf{x_b}) - H(\mathbf{x_a},\mathbf{x_b}) \\ &= I_{\mathbf{x_a};\mathbf{x_b}} \end{aligned} \tag{26}$$

Thus, the mutual information between the outputs, assuming the transformation computed by the network is invertible and of the same dimensionality as the input, is equal to the mutual information between the inputs. In other words, any function computed by the network that spans the same space as the input would be equivalent. The maximization of mutual information in this case has an infinite number of solutions, many of which are degenerate. Obviously, the more interesting case for Imax is when the output dimension is smaller than that of the input. Thus, an approximation to the joint entropy of continuous variables is needed for this case.

3 Discussion

3.1 Computational complexity

Both discrete and continuous Imax between a pair of units can be inexpensively implemented. The complexity of each gradient computation is $O(n)$, where n is the number of output units. To compute the exact derivatives requires two passes through the set of training patterns, one to estimate the statistics and one to compute the gradients. These statistics can alternatively be approximated online to reduce the learning procedure to a single pass through the training set per learning iteration. Multivariate discrete Imax requires $O(n^2)$ operations, and $O(n^2)$ storage, because each pair of units in the two modules must compute their joint probability of being active. In a real neural network, of course, these statistics would be computed in parallel by having connections between each pair of output neurons that store the joint probabilities and modulate the learning. Multivariate continuous Imax requires a rather daunting $O(n^3)$ statistics to compute the determinant of the covariance matrix for the two sets of units outputs. Apart from the latter algorithm, the real complexity consideration for Imax learning comes into the training time. In all of my simulations, Imax typically takes on the order of several thousand passes through the training set to converge.

3.2 Biological plausibility

We touched on the question of the biological plausibility of Imax learning in Section 2.1. The algorithm requires the estimation of correlational statistics for the joint probabilities that pairs of units are on together. These statistics can be estimated online as learning proceeds. However, the correlations are between pairs of *output* units that are not directly connected to one another, making the algorithm somewhat implausible. Phillips et al. [17] suggested a way of making the computation of mutual information more biologically plausible: instead of having non-driving links between two units maximizing binary mutual information, one unit's activity could modulate that of the other unit. The modulated output, averaged over time, would represent the correlation between the features extracted from the two channels. A related algorithm proposed by de Sa [24] minimizes the disagreement error between the outputs of two modules. de Sa and Ballard [19] show how this learning mechanism could be carried out using only local computations, by adding a layer of competitive learning units on top, and having the competitive layer send modulatory feedback connections to the intermediary "minimizing disagreement" layer. When two units receiving input from different channels become correlated, they would tend to activate the same competitive unit and thereby receive reinforcing feedback; these units would therefore move their weights closer to their current inputs, while other units would move further away.

3.3 Alternatives to explicit mutual information maximization

The continuous Imax algorithm implicitly embodies a prior assumption about the coherence of the input signal: it assumes the input contains a signal that is approximately equal across the input channels, embedded in some other signal treated as noise. Yuille, Smirnakis and Xu [37] present one way to generalize this notion, so that it could be applied to other types of coherence as well. They propose minimizing the divergence between the network's representation of some underlying signal and a prior distribution of the signal. By appropriate choice of the prior, particular coherence constraints can be enforced. For example, in one special case, assuming a prior in which the signals extracted from neighboring receptive fields are equal and Gaussian, their cost function is equivalent to continuous Imax.

We have discussed a number of ways of generalizing Imax to allow for the extraction of multiple features. One drawback to continuous multivariate versions of Imax is that they involve the computation of the determinant. This is highly non-local, requiring higher-order statistics involving combinations of more than two units. The discrete version of Imax does not suffer from this problem, but its representation is less general. Discrete Imax assumes the output represents a classification of the input into one of n classes. I have proposed a learning procedure called Joint Probability Maximization (JPMAX) for discrete networks [38]. As in Yuille et al.'s model, the goal is to fit the behavior of the network to match a prior model. As in discrete Imax, but unlike Yuille's model, the JPMAX cost function depends on the *ensemble-averaged* joint probability distribution (output correlations) of two modules receiving input from different channels. An arbitrary prior model can be chosen for the joint probability distribution, for example, with a more distributed representation such as a population code. This allows for a compromise between the winner-take-all representation adopted by discrete Imax networks and the fully general (but computationally expensive) case of multivariate continuous Imax networks. I have applied JPMAX to the task of classifying images of coherently moving objects. It is able to perform early feature extraction in intermediate layer units, as well as pattern classification in output layer units. It can thus learn in several stages to do viewpoint-independent recognition of coherently moving objects, without the need to back-propagate derivatives.

To conclude, we have demonstrated that Imax is a powerful learning procedure for training neural networks to extract nonlinear features without external supervision. The idea of maximizing mutual information across different input channels has now been applied successfully to a number of different problems, described here and elsewhere [35, 1, 21, 30, 16]. Our work has been influential in the development of several related learning procedures [39, 31, 38, 37, 17, 24, 27]. While the direct optimization of mutual information in neural networks is computationally expensive, we have shown that the computation is feasible in a number of special cases. Phillips et al. [17] and de Sa and Ballard [19] have proposed unsupervised learning rules that are similar in spirit to Imax, but are much more biologically plausible. Thus, the general idea of extracting features that are coherent across different input channels is a promising way of deriving powerful, biologically plausible models of self-organization.

Acknowledgements

This work was supported by research grants from the James S. McDonnell Program in Cognitive Neuroscience and the Natural Sciences and Engineering Research Council of Canada. I am very grateful to Peter Dayan, Jim Stone, Ginny de Sa and Jim Kay for their helpful feedback on earlier drafts of this manuscript.

References

[1] S. Becker and G. E. Hinton. A self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355:161–163, 1992.

[2] P. J. B. Hancock, L. S. Smith, and W. A. Phillips. A biologically supported error-correcting learning rule. *Neural Computation*, 3:201–212, 1991.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by back-propagating errors. *Nature*, *323*:533–536, 1986.

[4] R. Linsker. Self-organization in a perceptual network. *IEEE Computer,* March, 21:105–117, 1988.

[5] H. B. Barlow. Cognitronics: Methods for acquiring and holding cognitive knowledge, October 1985. Unpublished manuscript.

[6] H. B. Barlow. Unsupervised learning. *Neural Computation*, 1:295–311, 1989.

[7] J. J. Atick and A. N. Redlich. Towards a theory of early visual processing. *Neural Computation*, 2:308–320, 1990.

[8] W. Bialek, D. L. Ruderman, and A. Zee. Optimal sampling of natural images: A design principle for the visual system? In *Advances In Neural Information Processing Systems 3*, pages 363–369. Morgan Kaufmann Publishers, 1991.

[9] M. D. Plumbley. Efficient information transfer and anti-hebbian neural networks. *Neural Networks*, 6(6):823–833, 1993.

[10] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423,623–656, 1948.

[11] J. J. Atick. Could information theory provide an ecological theory of sensory processing? *Network*, 3:213–152, 1992.

[12] S. B. Laughlin. Form and function in retinal processing. *Trends in neurosciences*, 10:478–483, 1987.

[13] S. Becker. *An Information-theoretic Unsupervised Learning Algorithm for Neural Networks*. PhD thesis, University of Toronto, 1992. Also published as technical report CRG-TR92-3.

[14] L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer. A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(7):1001–1008, 1989.

[15] P. F. Brown, V. J. Della Pietra, P. V. de Souza, J. C. Lai, and R. L. Mercer. Class-based n-gram models of natural language. In *Proceedings of the IBM Natural Language ITL*, pages 283–298, 1990.

[16] S. Becker and G. E. Hinton. Using spatial coherence as an internal teacher for a neural network. In Y. Chauvin and D. E. Rumelhart, editors, *Back-Propagation:Theory, Architectures, and Applications*, pages 313–349. Lawrence Erlbaum, Hillsdale, NJ., 1995.

[17] W. A. Phillips, J. Kay, and D. Smyth. The discovery of structure by multi-stream networks of local processors with contextual guidance. *Network*, 6:225–246, 1995.

[18] J. Kay. Probabilistic modelling of a binary multiple output local processor: Local objective functions, local learning rules and approximations based on information theoretic ideas. Unpublished manuscript, 1995.

[19] V. R. de Sa and D. Ballard. Neural networks for different modalities to teach eachother. *Document in preparation*, 1995.

[20] J. S. Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In D. S. Touretzky, editor, *Advances In Neural Information Processing Systems, Vol. 2*, pages 111–217, San Mateo, CA, 1990. Morgan Kaufmann.

[21] S. Becker. Learning to categorize objects using temporal coherence. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*, pages 361–368, San Mateo, CA, 1993. Morgan Kaufmann.

[22] S. Edelman. Representation of similarity in three-dimensional object discrimination. *Neural Computation*, 7:408–423, 1995.

[23] G. E. Peterson and H. L. Barney. Control methods used in a study of vowels. *Journal of the Acoustical Society of America*, 24:175–184, 1952.

[24] V. R. de Sa. Learning classification with unlabeled data. In *Advances in Neural Information Processing Systems 6*, pages 112–119. Morgan Kaufmann, 1994.

[25] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Proceedings of the Royal Statistical Society*, B-39:1–38, 1977.

[26] S. J. Nowlan. Maximum likelihood competitive learning. In D. S. Touretzky, editor, *Neural Information Processing Systems, Vol. 2*, pages 574–582, San Mateo, CA, 1990. Morgan Kaufmann.

[27] Z. Ghahramani. *Computation and Psychophysics of Sensorimotor Integration.* PhD thesis, MIT, Department of Brain and Cognitive Sciences, 1995.

[28] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis.* Academic Press, 1979.

[29] J. Kay. Feature discovery under contextual supervision using mutual information. In *Proceedings of the International Joint Conference on Neural Networks*, volume IV, pages 79–84, 1992.

[30] S. Becker and G. E. Hinton. Learning mixture models of spatial coherence. *Neural Computation*, 5(2):267–277, 1993.

[31] J Stone. Learning perceptually salient visual parameters using spatio-temporal smoothness constraints. to appear in *Neural Computation*, 1996.

[32] A. Ukrainec and S. Haykin. A mutual information-based learning strategy and its application to radar. In *Fuzzy Logic and Neural Network Handbook.* McGraw Hill, 1995.

[33] A. Ukrainec and S. Haykin. A modular neural network for enhancement of cross-polar radar targets. to appear in *Neural Networks*, 1996.

[34] S. Haykin. Polarimetric radar for accurate navigation. *Canadian Journal of Electrical and Computer Engineering*, 17:130–135, 1992.

[35] R. S. Zemel and G. E. Hinton. Discovering viewpoint-invariant relationships that characterize objects. In R. P. Lippmann, J. E. Moody, and D. S. Touretzky, editors, *Advances In Neural Information Processing Systems 3*, pages 299–305. Morgan Kaufmann Publishers, 1991.

[36] T. J. Bell and T. J. Sejnowski. An information-maximisation approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[37] A. L. Yuille, M. S. Stelios, and L. Xu. Bayesian self-organization driven by prior probability distributions. *Neural Computation*, 7:580–593, 1994.

[38] S. Becker. JPMAX: Learning to recognize moving objects as a model-fitting problem. In *Advances in Neural Information Processing Systems 7*, pages 933–940. MIT Press, 1995.

[39] A. Ukrainec and S. Haykin. Application of unsupervised neural networks to the enhancement of polarization targets in dual-polarized radar images. In *IEEE Canadian Confrernce on Elecrtical and Computer Engineering*, 1991.

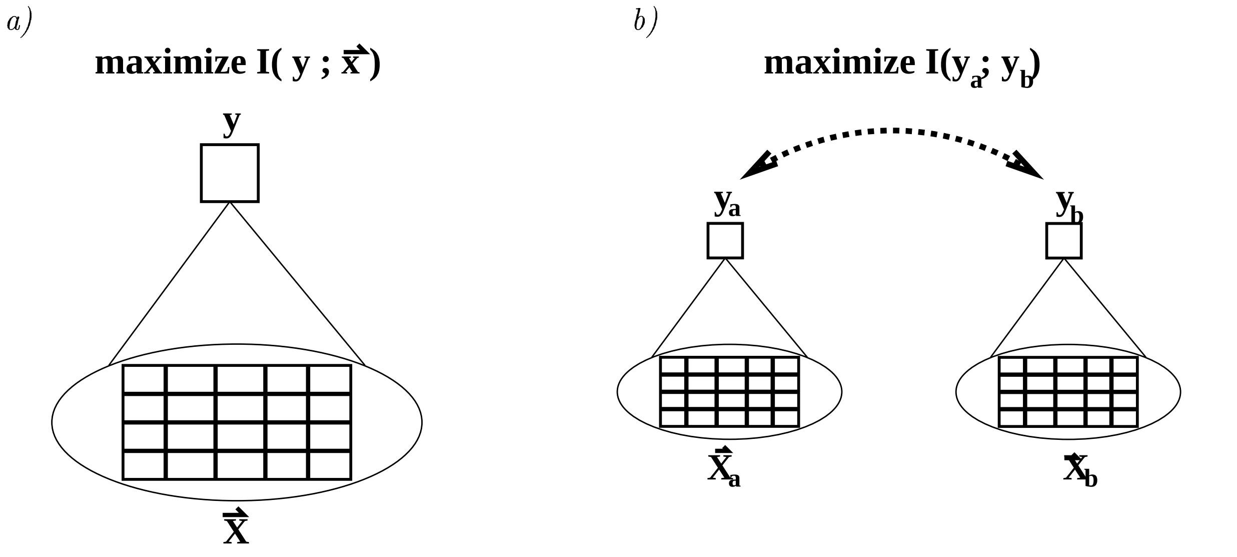

Figure 1: *a) An information transmission model of a neuron. The mutual information between the neuron's input and output $I(\vec{x}; y)$, or some measure involving I, is optimized. b) An Imax model in which the goal is to maximize the mutual information between the two output neuron's responses.*

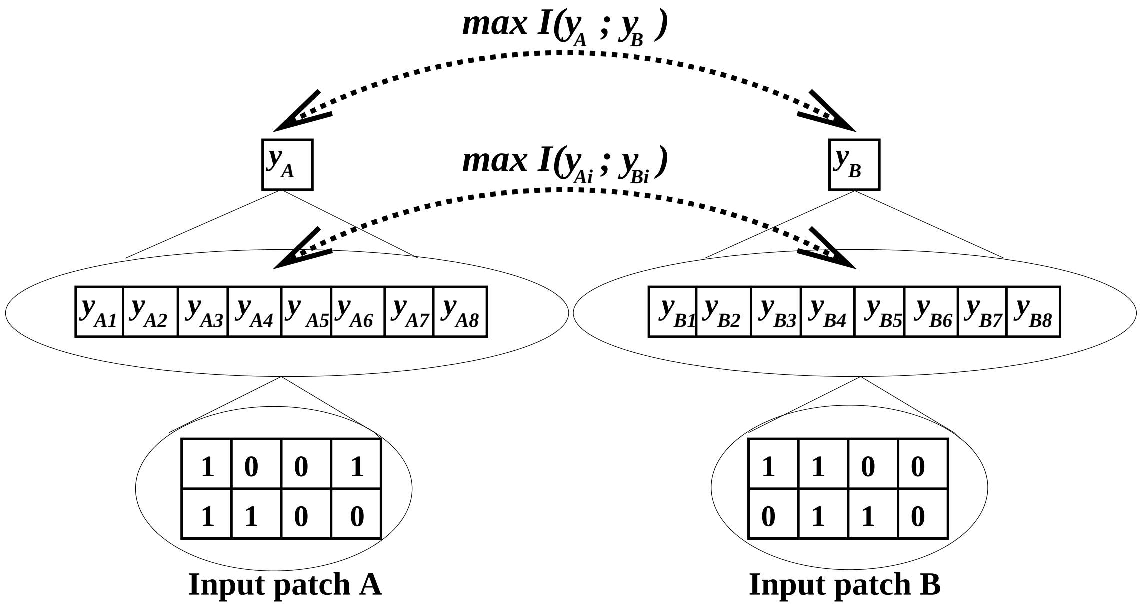

Figure 2: *The modular architecture used to solve the shift problem. Units in successive layers within modules are fully interconnected with feed-forward links. The Imax learning algorithm for binary units maximizes the mutual information between pairs of units' outputs in different modules. An example of a binary right-shifted pattern is shown.*

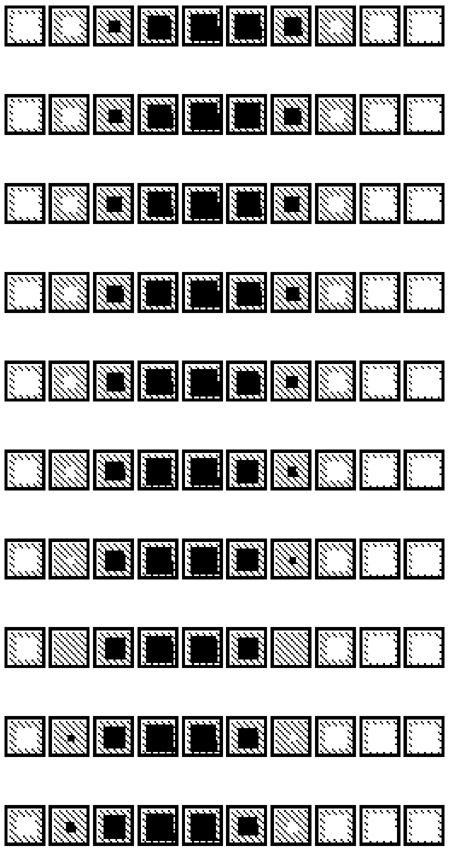

Figure 3: *Ten successive input patterns to the network, out of a training set of 1800 patterns. Each row of ten squares represents the states of the ten input units for a single training pattern. Black squares are negative and white are positive. The size of the square is proportional to the activity level of the unit. At each time step, a new pattern is presented to the network, and the sine wave drifts a small distance. The pattern shifts one complete cycle after every 50 patterns or time steps, at which point the frequency changes. The data set contained patterns of three frequencies: .1, .2 and .3 cycles per pixel. The pattern shown here has a frequency of 0.1 cycles/pixel.*

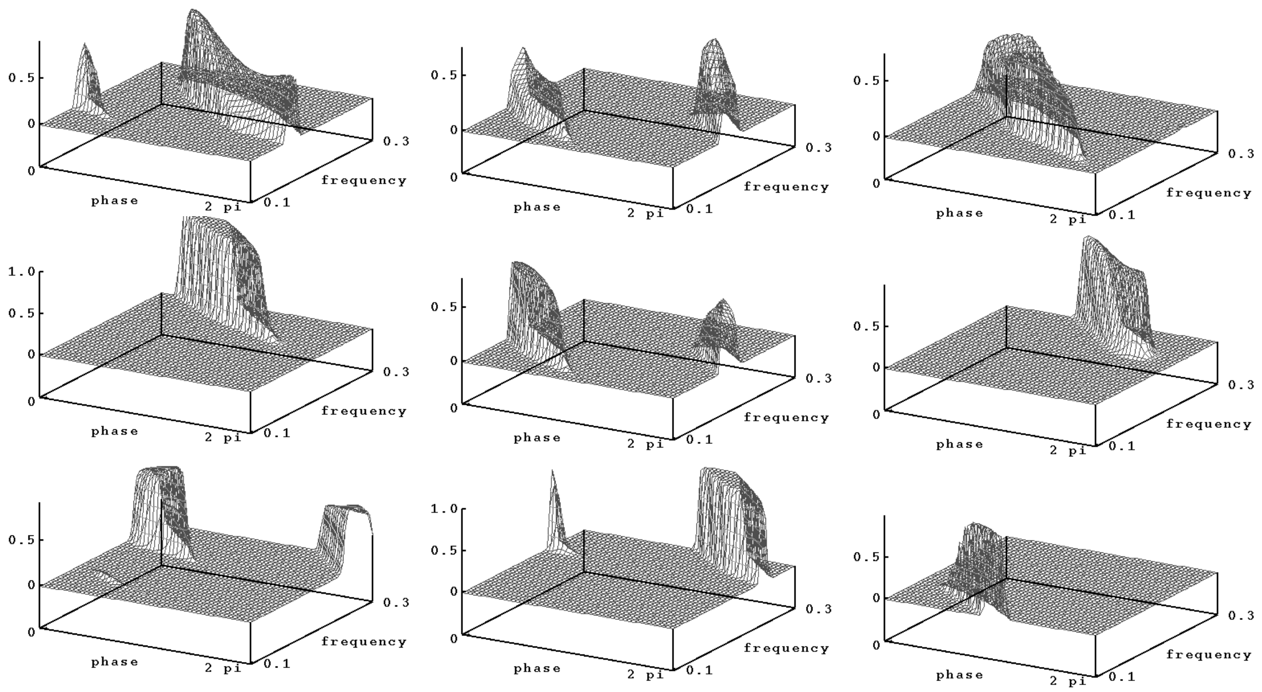

Figure 4: *Frequency-phase tuning curves of the first nine of twenty units in the first layer. The layer of units learned by maximizing the discrete mutual information between its outputs at successive time steps. Each graph shows a plot of a single first layer unit versus spatial frequency and phase. To generate these plots, a set of test patterns was used that had continuously varying spatial frequencies between 0.1 and 0.3 cycles per pixel. As in the training set, phase varied continously between 0 and 2 pi radians.*

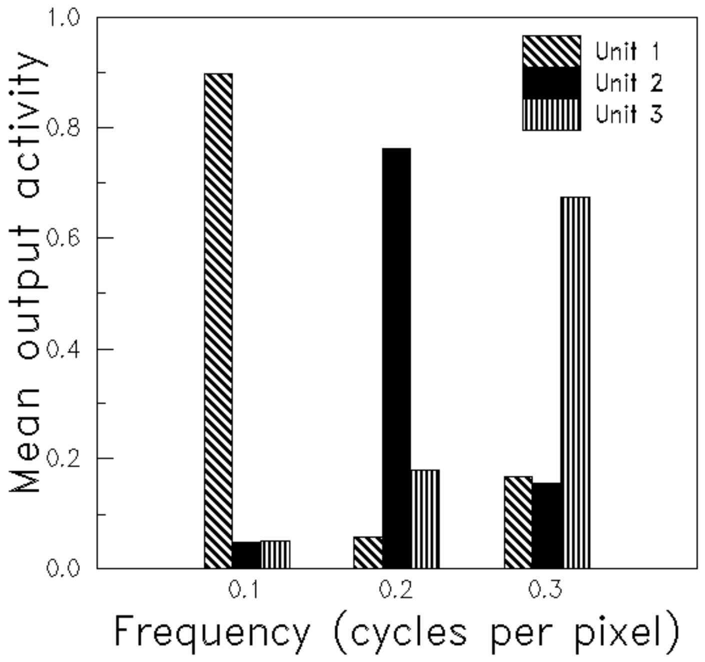

Figure 5: *Activity histograms of the three units in the second layer, plotted against spatial frequency. The layer of units learned by maximizing the mutual information bewteen its outputs at neighboring time steps.*

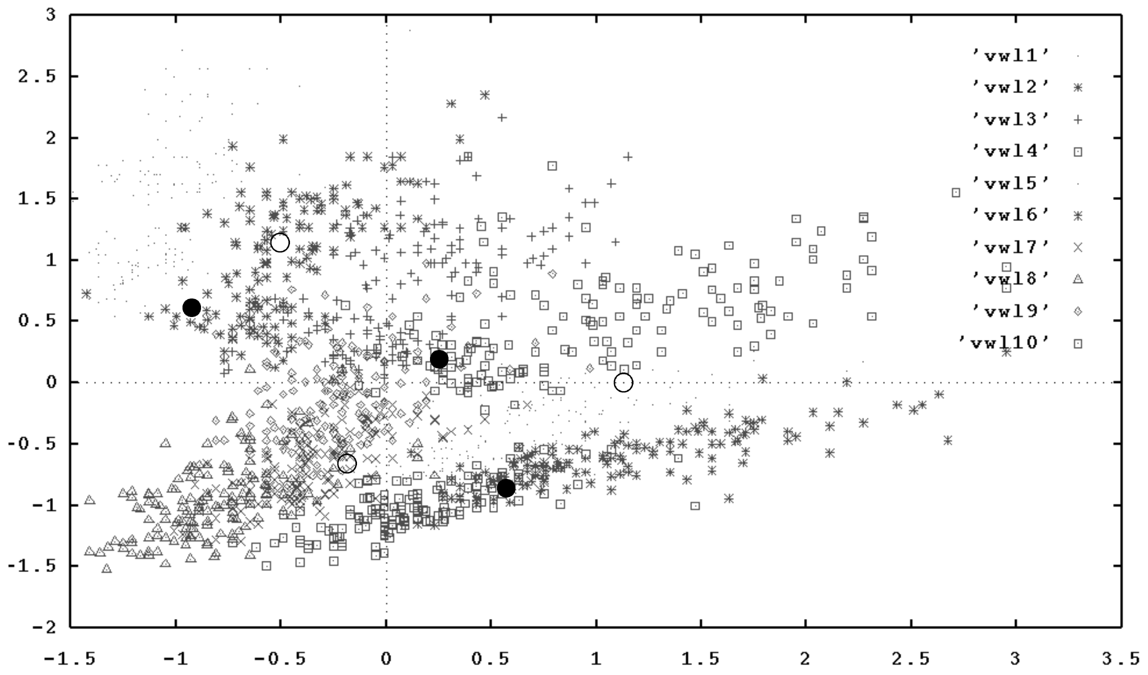

Figure 6: *The Peterson-Barney vowel data, consisting of the first and second formant of ten vowels, spoken by a variety of male and female speakers. Only widely separated pairs of classes are displayed with the same symbols. The three Gaussian centers learned by Imax are shown with filled circles, and those learned by EM1 are shown with open circles.*

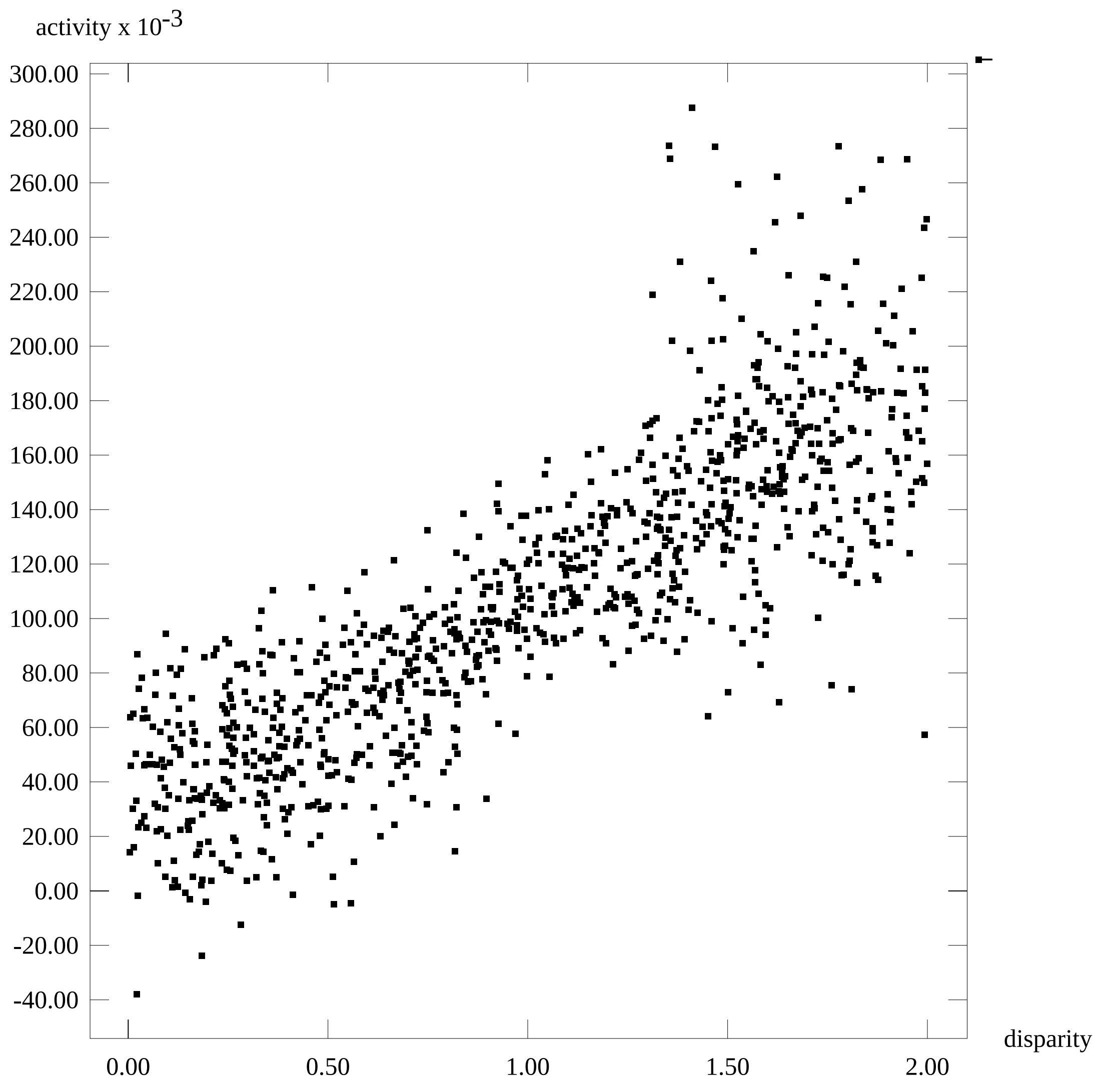

Figure 7: *The output of a neuron as a function of disparity (in pixels), for a network trained with 10 adaptive nonlinear hidden units and one linear output unit per module on random dot stereograms of planar surfaces. The network learned by maximizing the continuous mutual information between the outputs of neighboring modules.*