

Chapter 4

Information theoretic approaches to neural network learning

M D Plumbley
Kings College London, UK

4.1 Introduction

It is now nearly 50 years since Shannon first formulated his ‘Mathematical theory of communication’ [21], where he first introduced *information theory*, which has been used and developed extensively by communications engineers ever since. Communications engineering is concerned with transmitting information from one place to another as efficiently as possible, given certain costs and constraints which are imposed on the communications system which we wish to use. For example, we may have a maximum number of *bits* (binary digits) per second that we can send down a certain binary transmission link, or we may have a radio transmitter with a limit on the maximum power level which we can use. Either of these define constraints which we must work within.

Shannon’s information theory gave communication engineering a precise meaning to the idea of *rate of information transmission*. This helped to explain how the properties of a communication channel can limit how fast information can be transmitted through it, and how to *code* signals to make most efficient use of such a channel. One of these results showed that a channel has an innate limit on its information rate, its *capacity*. It is impossible to send information through a channel faster than that channel’s *capacity*, no matter how the information is represented or coded.

Other results from information theory deal with *noise* in a communication channel, which may disrupt or degrade a signal. In particular, the presence of noise reduces the information capacity of a

channel. However, it is possible to use *error-correcting codes* to reduce the probability of an error in a noisy channel to as close to zero as we like.

Almost as soon as information theory first appeared, psychologists and physiologists were interested in the idea that information theory could help to explain the mechanisms of perception. The visual system of a living creature, for example, could be transmitting information in some form to higher centres of the brain. Treating this visual system as a communications system might help us to understand some of the details behind the function it performs.

Early on, Attneave [4] suggested that visual perception is the construction of an economical description of a scene. Using a guessing game to measure information, he suggested that information in a visual scene is concentrated around the edges and corners of an image, since they are the least predictable from their surroundings. This is consistent with the structural arrangement of simple cells now known to exist in the visual cortex.

More recently, with the resurgence of the field of neural networks, authors such as Linsker [12], Barlow and Földiák [6], Plumbley and Fallside [19], and Atick and Redlich [1] have continued the use of information theory in neural networks, with considerable success. Information theory has proved particularly useful in the development of *unsupervised* learning algorithms. Unlike supervised learning algorithms such as Error Back-Propagation ('BackProp'), unsupervised algorithms have no 'teacher' output to specify what the output of the network should produce. It is therefore more difficult to determine what task or function such a network should learn to perform.

This paper is organized as follows. Section 4.2 gives a brief introduction to information theory, and introduces Linsker's *Infomax* principle, and information loss. Section 4.3 shows how networks which perform *principal components analysis* (PCA) can be viewed from this information-theoretic perspective, by making assumptions about the type of input noise. Section 4.4, shows how inhibitory interneurons can be used in this framework, when there is noise on the output of the network. Finally, section 4.5 introduces Becker and Hinton's *I-Max* which extracts depth information from stereograms by maximizing information between neighbouring image patches.

4.2 Information theory

4.2.1 Entropy and information

The two central concepts of information theory are those of *entropy* and *information* [21]. Generally speaking, the entropy of a set of outcomes is the uncertainty of our knowledge about which outcome will actually happen: the less sure we are about the outcome, the higher the entropy. If we know for sure what the outcome will be, the entropy will be zero.

Information is gained by reducing entropy, for example by making an observation of an outcome. Before the observation, our knowledge of the outcome is limited, so we have some uncertainty about it. However, after the observation the entropy (uncertainty) is reduced to zero: the difference is the information gained by the observation. A concrete example will be given in a moment, after we introduce the formulas for entropy and information.

Consider an experiment with N possible outcomes i , $1 \leq i \leq N$ with respective probabilities p_i . The entropy H of this system is defined by the formula

$$H = - \sum_{i=1}^N p_i \log p_i \quad (4.1)$$

with $p \log p$ equal to zero in the limit $p = 0$.

For example, for a fair coin toss, with $N = 2$ and $p_1 = p_2 = 1/2$, we have

$$\begin{aligned} H &= - \left(\frac{1}{2} \log \frac{1}{2} + \frac{1}{2} \log \frac{1}{2} \right) \\ &= \log 2 \end{aligned}$$

If the logarithm is taken to base 2, this quantity is expressed in ‘bits’, so a fair coin toss has an entropy of 1 bit.

For any number of outcomes N , the entropy is maximized when all the probabilities are equal to $1/N$. In this case, the entropy is $\log N$. If one of the outcomes has probability 1 with all others having probability 0, then the entropy H in (4.1) is zero: otherwise, H is always positive.

As we mentioned before, the information gained by an observation is the entropy before it, less the entropy after it. As an example, consider our coin toss again, and assume that we observe the outcome to be a ‘head’. We denote the state of the coin by the random variable Ω , and write the entropy of the coin toss before any observation by $H(\Omega)$ (figure 4.1(a)). If we denote the observed face by X , we write the *conditional entropy* of the coin after the observation as $H(\Omega|X = \text{‘head’})$ (figure 4.1(b)).

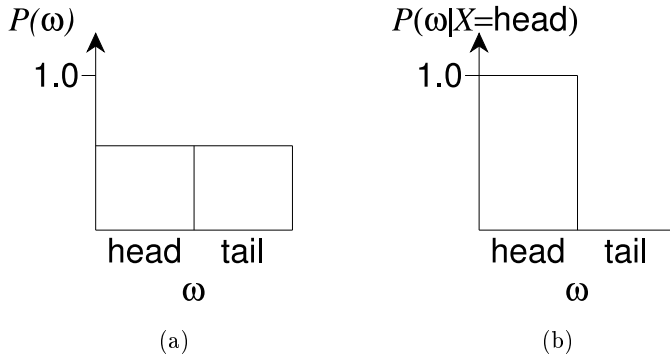


Figure 4.1. Probabilities of coin state $\Omega = \omega$ for (a) before observation, and (b) after observation of a ‘head’.

The situation if the outcome is a ‘tail’ is exactly the same. The information in the observation X about the coin state Ω is then written

$$I(\Omega, X) = H(\Omega) - H(\Omega|X) = \log 2$$

i.e. one bit of information was gained by the observation.

For continuous variables, we cannot use the original discrete-variable formula (4.1), since we now have an infinite number of possible states, which would lead to infinite entropy. Instead we use an alternative form

$$H = - \int_{-\infty}^{\infty} p(x) \log p(x) dx \quad (4.2)$$

which is normally finite, but no longer guaranteed to be positive, and is also dependent on the scaling of variables: scaling by n will add $\log n$ to the entropy.

The information $I(\Omega, X) = H(\Omega) - H(\Omega|X)$ derived from this continuous case *is* scale independent, however (figure 4.2), since the scaling will add the same value to both ‘before’ and ‘after’ entropies. The entropy $H(\Omega|X)$ represents the noise in the observation X . As an example, for a Gaussian signal of variance $\sigma_S^2 = S$ and noise of variance $\sigma_N^2 = N$, we can calculate that the mean information gained from an observation is

$$I = 0.5 \log(1 + S/N)$$

where S/N is the signal to noise power (variance) ratio. As the noise power N goes to zero, the information gained becomes infinite: so if we could

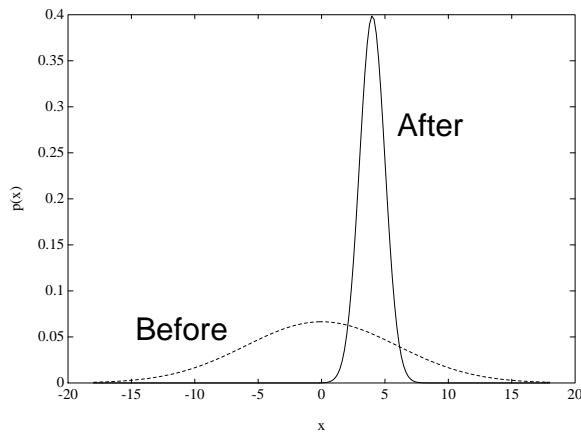


Figure 4.2. Probabilities of a Gaussian distribution before and after a noisy observation. The ‘before’ distribution has the signal entropy $H(\Omega)$, while the ‘after’ distribution has the noise entropy $H(\Omega|X)$ for an example observation $X = 4$.

measure a continuous quantity with complete accuracy, we would gain an infinite amount of information. Consideration of noise is therefore very important when determining the information available from a continuous value.

4.2.2 Infomax and information loss

If we think of the early parts of a perceptual system such as vision as a system for transmitting information about the environment on to higher centres, it seems reasonable that the more information which is transmitted, the more effective the system will be. Of course, some visual systems are optimized to extract information about very specific stimuli early on: an example would be the ‘bug detectors’ found in the frog [11]. For higher animals, however, it is more likely that early parts of the visual system should process all input information equally. Linsker therefore suggested his *Infomax* principle: that a perceptual system should attempt to organize itself to maximize the rate of information transmitted (in bits per second) through the system [12].

An alternative view introduced by Plumley and Fallside [19] is to try to *minimize* the *loss* in information about some original signal Ω as

the sensory input is processed by the perceptual system or neural network. Although this approach is in many ways equivalent to Linsker's Infomax principle, it allows a minimax approach to be used in cases when the signal is not Gaussian, for example.

Information loss about Ω across the system which transforms X to Y (figure 4.3) is denoted by

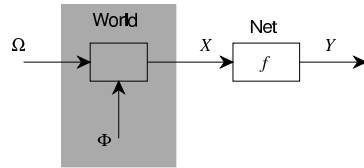


Figure 4.3. The original signal Ω is corrupted by irrelevant noise Φ to give the stimulus X . This is then transformed by the network function f to give the output Y .

$$\Delta I_{\Omega}(X, Y) = I(X, \Omega) - I(Y, \Omega) \quad (4.3)$$

and has the following properties:

- (i) ΔI is positive across any function f , such that $Y = f(X)$;
- (ii) ΔI is positive across any additive noise Φ , such that $Y = X + \Phi$ (figure 4.4(a)).
- (iii) ΔI is additive across a chain of networks (figure 4.4(b)).

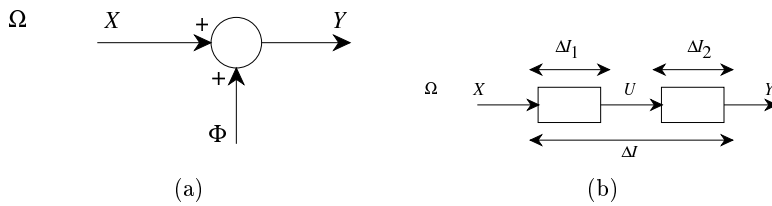


Figure 4.4. Information loss is (a) positive across additive noise, and (b) additive in series.

So, to minimize the information loss across a series of networks, the information loss across each network should be minimized. Once information is lost, it cannot be regained.

For an example of this approach, in the next section we examine networks which extract the *principal components* of the inputs. These are often used as a first processing stage, to simplify a pattern recognition problem by reducing the number of dimensions. We shall see what happens when we try to optimize the information transmitted across this type of network.

4.3 Principal components analysis

Principal component analysis (PCA) is widely used for dimension reduction in data analysis and pre-processing, and is used under a variety of names such as the (discrete) Karhunen Loève Transform (KLT), factor analysis, or the Hotelling Transform in image processing. Its primary use is to provide a reduction in the number of parameters used to represent a quantity, while minimising the error introduced by doing so. In the case of PCA, a purely linear transform is used to reduce the dimensionality of the data, while minimising the mean squared reconstruction error. This is the error which we get if we transform the output y back into the input domain to try to reconstruct the input \mathbf{x} so that the error is minimised.

Linsker's principal of maximum information preservation, Infomax, can be applied to a number of different forms of neural network. The analysis, however, is much simpler when we are dealing with simple networks, such as binary or linear systems. It is instructive to look at the linear case of PCA in some detail, since much effort in other fields has been directed at linear systems. We should not be too surprised to find a neural network system which can perform KLT and PCA.

From one point of view, these conventional data processing methods let us know what to expect from a linear unsupervised neural network. However, the information theoretic approach to the neural network system can help us with conventional data processing methods. In particular, we shall find that a dilemma in the use of PCA, known as the *scaling problem*, can be clarified with the help of information theory.

4.3.1 The linear neuron

Arguably the simplest form of unsupervised neural network is an N -input, single-output linear neuron (figure 4.5). Its output response y is simply the sum of the inputs x_i multiplied by their respective weights w_i , i.e.

$$y = \sum_{i=1}^N w_i x_i \quad (4.4)$$

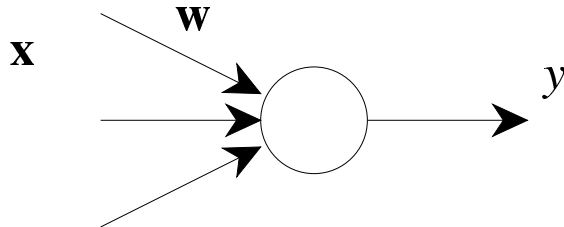


Figure 4.5. The Oja Neuron.

or, in vector notation,

$$y = \mathbf{w}^T \mathbf{x} \quad (4.5)$$

where $\mathbf{w} = [w_1, \dots, w_N]^T$ and $\mathbf{x} = [x_1, \dots, x_N]^T$ are column vectors. The output y is thus the dot product $\mathbf{x} \cdot \mathbf{w}$ of the input \mathbf{x} with the weight vector \mathbf{w} . If \mathbf{w} is a unit vector, i.e. $|\mathbf{w}| = 1$, y is the component of \mathbf{x} in the direction of \mathbf{w} (figure 4.6).

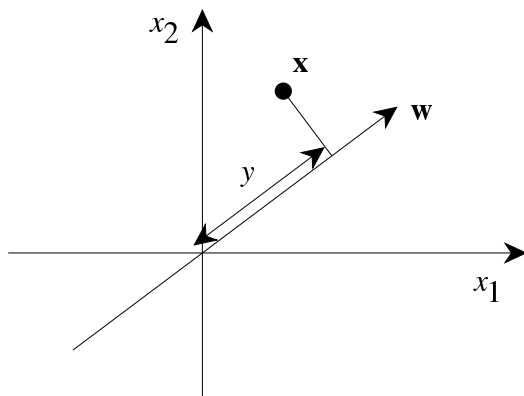


Figure 4.6. Output y as a component of \mathbf{x} , with unit weight vector.

We thus have a simple neuron which finds the component of the input \mathbf{x} in a particular direction. We would now like to have a neural network

learning rule for this system, which will modify the weight vector depending on the inputs which are presented to the neuron.

4.3.2 The Oja principle component finder

Many simple learning rules are based on Hebb's principle [10], which states that the effectiveness of a connection between two neurons should be increased when they are both active at the same time. A very simple form of Hebbian learning rule for the connections in this neuron would be to update each weight by the product of the activations of the units at either end of the weight. For the single linear neuron (figure 4.5), this would result in a learning algorithm of the form

$$\Delta w_i = \eta x_i y \quad (4.6)$$

or in vector notation

$$\Delta \mathbf{w} = \eta \mathbf{x} y \quad (4.7)$$

where η is a learning rate parameter. Unfortunately, this learning algorithm alone would cause any weight to increase without bound, so some modification has to be used to prevent the weights from becoming too large.

One possible solution is to limit the absolute values that each weight w_i can take [23], while another is to renormalise the weight vector \mathbf{w} to have unit length after each update [13]. An alternative is to use a weight decay term which causes the weight vector to tend to have unit length as the algorithm progresses, without explicitly normalising it. To see how this works, consider the following weight update algorithm, due to Oja [13]:

$$\begin{aligned} \Delta \mathbf{w} &= \eta(\mathbf{x} y - \mathbf{w} y^2) \\ &= \eta(\mathbf{x} \mathbf{x}^T \mathbf{w} - \mathbf{w}(\mathbf{w}^T \mathbf{x} \mathbf{x}^T \mathbf{w})). \end{aligned} \quad (4.8)$$

When the weight vector is small, the update algorithm is dominated by the first term on the right hand side, which causes the weight to increase as for the unmodified Hebbian algorithm. However, as the weight vector increases, the second term (the 'weight decay' term) on the right hand side becomes more significant, and this tends to keep the weight vector from becoming too large.

To find the convergence conditions of the Oja algorithm, let us consider the average weight update over some number of input presentations. We shall assume the input vectors \mathbf{x} have zero mean, and we shall also assume that the weight update factor is so small that the weight itself can be

regarded as approximately constant over this number of presentations. Thus the mean update is given by

$$\begin{aligned}\overline{\Delta \mathbf{w}} &= \eta \left(\overline{\mathbf{x} \mathbf{x}^T \mathbf{w}} - \overline{\mathbf{w} \mathbf{w}^T \mathbf{x} \mathbf{x}^T \mathbf{w}} \right) \\ &\approx \eta \left(\overline{\mathbf{x} \mathbf{x}^T \mathbf{w}} - \mathbf{w} \left(\overline{\mathbf{w}^T \mathbf{x} \mathbf{x}^T \mathbf{w}} \right) \right) \\ &\approx \eta (\mathbf{C}_x \mathbf{w} - \mathbf{w} \lambda)\end{aligned}\quad (4.9)$$

where $\lambda = \mathbf{w}^T \mathbf{C}_x \mathbf{w}$, and $\mathbf{C}_x = E(\mathbf{x} \mathbf{x}^T)$ is the covariance matrix of the input data \mathbf{x} .

When the algorithm has converged, the average value of $\Delta \mathbf{w}$ will be zero, so we have

$$\mathbf{C}_x \mathbf{w} = \mathbf{w} \lambda \quad (4.10)$$

i.e. the weight vector \mathbf{w} is an eigenvector of the input covariance matrix \mathbf{C}_x . A perturbation analysis confirms that the only stable solution is for \mathbf{w} to be the *principal* eigenvector of \mathbf{C}_x . To find the eventual length of \mathbf{w} we simply substitute (4.10) into the expression for λ , and we find that

$$\lambda = \mathbf{w}^T (\mathbf{C}_x \mathbf{w}) = \mathbf{w}^T (\mathbf{w} \lambda) \quad (4.11)$$

i.e. provided λ is non-zero, $\mathbf{w}^T \mathbf{w} = 1$ so the final weight vector has unit length.

We have therefore seen that as the Oja algorithm progresses, the weight vector will converge to the normalised principal eigenvector of the input covariance matrix (or its negative) [13]. The component of the input which is extracted by this neuron, to be transmitted through its output y , is called the *principal component* of the input, and is the component with largest variance for any unit length weight vector.

4.3.3 Reconstruction error

For our single-output system, suppose we wish to find the best estimate $\hat{\mathbf{x}}$ of the input \mathbf{x} from the single output $y = \mathbf{w}^T \mathbf{x}$. We form our reconstruction using the vector \mathbf{u} as follows:

$$\hat{\mathbf{x}} = \mathbf{u} y \quad (4.12)$$

where \mathbf{u} is to be adjusted to minimise the mean squared error

$$\epsilon = E \left[|\mathbf{x} - \hat{\mathbf{x}}|^2 \right] = E \left[|(\mathbf{I} - \mathbf{u} \mathbf{w}^T) \mathbf{x}|^2 \right] \quad (4.13)$$

where \mathbf{I} is the identity matrix. If we minimise ϵ with respect to \mathbf{u} for a given weight vector \mathbf{w} , we get a minimum for ϵ at

$$\mathbf{u} \mathbf{w} = \arg \min_{\mathbf{u}} (\epsilon) = \frac{\mathbf{C}_x \mathbf{w}}{\mathbf{w}^T \mathbf{C}_x \mathbf{w}} \quad (4.14)$$

where $\mathbf{C}_x = E[\mathbf{x}\mathbf{x}^T]$ as before (assuming that \mathbf{x} has zero mean). Our best estimate of \mathbf{x} is then given by

$$\hat{\mathbf{x}}_w = \mathbf{u}_w y = \frac{\mathbf{C}_x \mathbf{w} \mathbf{w}^T}{\mathbf{w}^T \mathbf{C}_x \mathbf{w}} \mathbf{x} = \mathbf{Q} \mathbf{x} \quad (4.15)$$

where the matrix

$$\mathbf{Q} = \frac{\mathbf{C}_x \mathbf{w} \mathbf{w}^T}{\mathbf{w}^T \mathbf{C}_x \mathbf{w}} \quad (4.16)$$

is a *projection* operator, a matrix operator which has the property that $\mathbf{Q}^2 = \mathbf{Q}$. This means that the best estimate of the reconstruction vector $\hat{\mathbf{x}}_w$ from the output $\hat{\mathbf{y}}_w = \mathbf{w} \hat{\mathbf{x}}_w$ is $\hat{\mathbf{x}}_w$ itself. Once this is established, it is possible to minimise ϵ with respect to the original weight vector \mathbf{w} . Provided the input covariance matrix \mathbf{C}_x is positive definite, this minimum occurs when the weight vector is the principal eigenvector of \mathbf{C}_x . Thus PCA minimises mean squared reconstruction error.

4.3.4 The Scaling Problem

Users of PCA are sometimes presented with a problem known as the *scaling problem*. The result of PCA, and related transforms such as Karhunen Loève Transform (KLT), is dependent on the scaling of the individual input components x_i . When all of the input components come from a related source, such as light level receptors in an image processing system, then it is obvious that all the inputs should have the same scaling. However, when different inputs represent unrelated quantities, then the relative scaling which each input should be given is not so apparent. As an extreme example of this problem, consider two uncorrelated inputs which initially have equal variance. Whichever input has the largest scaling will become the principal component. While this extreme situation is unusual, the scaling problem does cause PCA to produce scaling-dependent results, which is rather unsatisfactory.

Typically, this dilemma is solved by scaling each input to have the same variance as each other [24]. However, there is also a related problem which arises when multiple readings of the same quantity are available. These readings can either be averaged to form a single reading, or they can be used individually as separate inputs. If same-variance scaling is used, these two options again produce inconsistent results.

Thus although PCA is used in many problem areas, these scaling problems may lead us not to trust it to give us a consistent result in an unsupervised learning system.

4.3.5 Information Maximization

We have seen that the Oja neuron learns to perform a principal component analysis of its input, but that principal component analysis itself suffers from an inconsistency problem when the scaling of the input components is not well defined. In order to gain some insight to this problem, Linsker applied his *Infomax* principle [12] to this situation.

Consider a system with input X and output Y . Linsker's Infomax principle states that a network should adjust itself so that the information $I(X, Y)$ transmitted to its output Y about its input X should be maximised. This is equivalent to the information in the input X about the output Y , since $I(X, Y)$ is symmetric in X and Y .

However, if Y is a noiseless function of X , as is the case for our linear neuron

$$y = \mathbf{w}^T \mathbf{x}$$

then there will be an *infinite* amount of information in the output Y about X , because Y represents X infinitely accurately. In order to proceed, we must assume that the input contains some noise ϕ which prevents any of the input from being measured too accurately.

Consider the case where the input signal \mathbf{x} is zero mean Gaussian with covariance matrix \mathbf{C}_x , and the noise ϕ is also zero mean Gaussian, with covariance matrix $\mathbf{C}_\phi = \sigma^2 \mathbf{I}$, so that the noise on each input component is uncorrelated with equal variance. The output of the neuron is then the weighted sum

$$y = \mathbf{w}^T (\mathbf{x} + \phi) = \mathbf{w}^T \mathbf{x} + \mathbf{w}^T \phi. \quad (4.17)$$

Writing down the information in the output y about the input signal \mathbf{x} , we get

$$I(Y; X) = \frac{1}{2} \log \frac{S + N}{N} \quad (4.18)$$

where

$$S = E(|\mathbf{w}^T \mathbf{x}|^2) = \mathbf{w}^T \mathbf{C}_x \mathbf{w}$$

and

$$N = E(|\mathbf{w}^T \phi|^2) = \sigma^2 |\mathbf{w}|^2.$$

Since (4.18) is monotonically increasing in S/N , $I(X, Y)$ is maximised when \mathbf{w} is the principal eigenvector of \mathbf{C}_x , i.e. it is the principal component of the input.

This is the same condition for minimising the mean squared reconstruction error considered above, but now we have an explicit condition on the noise on the input. The condition is that the noise on

each input should be uncorrelated, and each input should have the same noise variance.

The scaling problem of principal component analysis is now changed to one of guessing the noise on each of the inputs, and scaling them so that this noise is approximately equal. The standard approach of scaling all inputs so that their signal variance is equal is therefore equivalent to assuming that the signal to noise ratio of all inputs is equal [15].

Of course, the assumptions that the input signal and noise are Gaussian and zero mean are very strong, but can be relaxed somewhat if *information loss* is considered rather than *transmitted information*. However, the result in each case is the same: the Oja algorithm, which finds the principal component of the input, maximises information capacity on condition that the noise on the input components is equal.

4.3.6 Multi-dimensional PCA

There are a number of algorithms which extend Oja's algorithm to more than one output neuron. For these we need an output vector \mathbf{y} and a weight matrix \mathbf{W} , such that

$$\mathbf{y} = \mathbf{W}^T \mathbf{x}. \quad (4.19)$$

If the Oja algorithm was used for each output neuron with no modification, each would find the same principal component of the input data. Some mechanism must be used to force the outputs to learn something different from each other.

One possibility is to use a lateral inhibition network between the output neurons, which forces their outputs to be decorrelated [9]. An alternative is to modify the weight decay term of the Oja algorithm: this approach is used by William's Symmetric Error Correction (SEC) algorithm [25], Oja and Karhunen's M -output PCA algorithm [14], and Sanger's Generalised Hebbian Algorithm (GHA) [20].

In fact, these algorithms have much in common. Although the weight vectors themselves have different algorithms, the subspace defined by the orthogonal projection

$$\mathbf{P} = \mathbf{W}(\mathbf{W}^T \mathbf{W})^{-1} \mathbf{W}^T$$

which is the subspace spanned by the weight vectors to each of the outputs, moves in exactly the same way for each of these algorithms. Since this subspace, rather than the weight vector itself, determines the change in information transmitted through the network, these three algorithms tend to increase the transmitted information in exactly the same way. All three lead to a set of weight vectors which spans the same space as the largest principal eigenvectors of the input covariance matrix, which is sufficient to

maximise the transmitted information (under the equal noise conditions which we outlined above) [18].

The three algorithms differ only in the behaviour of the weight vectors themselves. In particular, the SEC algorithm [25] leads to weight vectors which are orthogonal and of unit length, but which have no particular relationship to the eigenvectors of the input covariance matrix. Oja and Karhunen's algorithm [14] uses a Gramm-Schmidt Orthogonalisation (GSO) approach to find the principal components themselves, in order. Sanger's algorithm [20] uses GSO in a slightly different way, but also finds the principal components in order.

4.3.7 Uncorrelated outputs

We have seen that linear neurons, with a modified form of Hebbian learning algorithm, can learn to find the principal component or principal subspace of their input data. We have also seen that this principal subspace maximises information capacity of the system, under the condition that the input components have uncorrelated, independent, equal-variance Gaussian noise on all of the components.

When we perform principal component analysis in practice, we also tend to use a set of outputs which are decorrelated, or at the very least not highly correlated with each other. The algorithms considered here also do this, but this does not seem to be required to maximise information capacity. We should only have to find the principal subspace of the input data: correlation between the output components should be irrelevant.

The puzzle here arises because we have neglected noise which may occur *after* the network which we are currently considering. This noise may be due to added noise or calculation errors in stages following the PCA network. In the next section, we shall see that decorrelated outputs tend to be better protected against later noise than outputs which are highly correlated [6]. It may be that real perceptual systems are able to take account of both noise sources at the same time.

4.4 Lateral inhibition and anti-Hebbian learning

In the networks we have considered so far, we have had feedforward connections from one layer to another. In this section, we shall consider networks with *lateral* connections, i.e. connections between units within the same layer.

Soon after Attneave's suggestion [4] that a visual system should create an economical description of a scene, Barlow [5] argued that *lateral inhibition* could be a possible mechanism to achieve this economical

description. This would involve inhibitory connections between neurons within a layer. Any signal which is common to many neurons would be reduced by this lateral inhibition effect. Uniform areas of the visual field would produce little output, while edges would produce a significant output from this layer. This lateral inhibition would reduce the amount of *redundant* information, i.e. information contained in more than one signal, and hence produce a more economical representation.

Neural network algorithms have been suggested which can learn to perform this redundancy reduction. These are sometimes called *anti-Hebbian*, since they cause the *inhibition* to increase if the neurons at either end are active.

4.4.1 Decorrelating algorithms

Barlow and Földiák [6] suggested a network with linear recurrent lateral inhibitory connections (Fig. 4.7(a)) with an anti-Hebbian local learning algorithm. This network is designed to *decorrelate* its outputs, i.e. to produce network outputs which are all uncorrelated with each other. Removal of correlations like this is sometimes known as *whitening*.

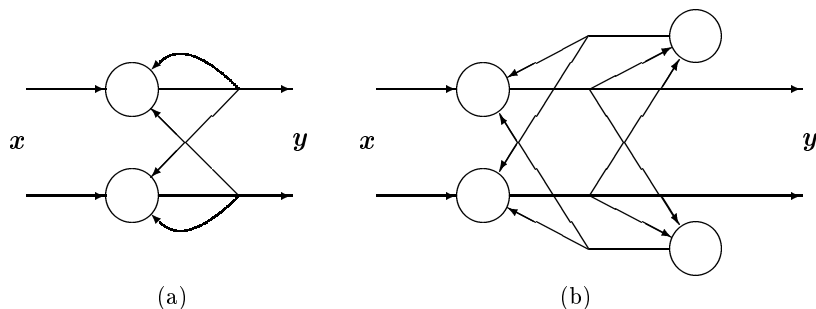


Figure 4.7. Linear decorrelating networks ($M = 2$).

In vector notation, we have an M -dimensional input vector \mathbf{x} , an M -dimensional output vector \mathbf{y} , and an $M \times M$ lateral connection matrix \mathbf{V} . For a fixed input, the lateral connections cause the output values to evolve according to the expression

$$(y_i)_{t+1} = x_i - \sum_j v_{ij}(y_j)_t \quad \text{i.e.} \quad \mathbf{y}_{t+1} = \mathbf{x} - \mathbf{V}\mathbf{y}_t \quad (4.20)$$

at time step t , which settles to an equilibrium when $\mathbf{y} = \mathbf{x} - \mathbf{V}\mathbf{y}$, which we

can write as

$$\mathbf{y} = (\mathbf{I}_M + \mathbf{V})^{-1} \mathbf{x} \quad (4.21)$$

provided $(\mathbf{I}_M + \mathbf{V})$ is positive definite. We assume that this settling happens virtually instantaneously. The matrix \mathbf{V} is assumed to be symmetrical so that the inhibition from unit i to unit j is the same as the inhibition from j to i , and for the moment we assume that there are no connections from a unit back to itself, so the diagonal entries of \mathbf{V} are zero.

Barlow and Földiák [6] suggested that for each input \mathbf{x} , the weights v_{ij} between different units should altered by a small change

$$\Delta v_{ij} = \eta y_i y_j \quad i \neq j \quad (4.22)$$

where η is a small update factor. In vector notation this is

$$\Delta \mathbf{V} = \eta \text{offdiag}(\mathbf{y} \mathbf{y}^T) \quad (4.23)$$

since the diagonal entries of \mathbf{V} remain fixed at zero. This algorithm converges when $E(y_i y_j) = 0$ for all $i \neq j$, and thus causes the outputs to become decorrelated [6].

Atick and Redlich [3] considered a similar network, but with an integrating output $d\mathbf{y}/dt = \mathbf{x} - \mathbf{V}\mathbf{y}$ leading to $\mathbf{y} = \mathbf{V}^{-1} \mathbf{x}$ when it has settled. They show that a similar algorithm for the lateral inhibitory connections between different output units leads to decorrelated outputs, while reducing a information-theoretic redundancy measure.

4.4.2 Optimizing information with lateral inhibition

The principal component algorithms discussed above were concerned with optimizing information given noise on the input to the network. These decorrelating algorithms instead attempt to optimize information in the case of noise on the *output* of the network.

To analyze this, we follow an argument by Shannon [22]. Consider a network with input represented by the random variable X , output Y , with added noise Φ giving a final information-bearing output $\Psi = Y + \Phi$.

For small output noise, we can express the transmitted information as

$$I(\Psi, X) = 1/2 \log \det \mathbf{C}_Y - 1/2 \log \det \mathbf{C}_\Phi \quad (4.24)$$

and the power cost as

$$S_T = \text{Tr}(\mathbf{C}_Y). \quad (4.25)$$

We wish to maximize $I(\Psi, X)$ for fixed S_T , so we use the Lagrange multiplier technique, and instead attempt to maximise the function

$$J = I(\Psi, X) - 1/2\lambda S_T. \quad (4.26)$$

This leads to the condition [16]

$$\mathbf{C}_Y = (1/\lambda)\mathbf{I}_M \quad (4.27)$$

so, not only should the outputs be decorrelated, but they should all have the same variance, $E(y_i^2) = 1/\lambda$.

The decorrelating algorithms outlined above will be sufficient if we know that the signal we are dealing with is not dependent on position. This might be the case for an image on a regular grid, for example, provided images could appear anywhere on this grid. If this is not the case, we may need a slightly different algorithm.

The Barlow and Földiák [6] algorithm can be modified to achieve this, if self-inhibitory connections from each unit back to itself are allowed [16]. The algorithm becomes

$$\Delta v_{ij} = \eta y_i y_j - (1/\lambda)\delta_{ij} \quad \text{i.e.} \quad \Delta \mathbf{V} = \eta(\mathbf{y}\mathbf{y}^T - (1/\lambda)\mathbf{I}_M) \quad (4.28)$$

which monotonically increases J as it progresses.

This is perhaps a little awkward, since the self-inhibitory connections have a different update algorithm to the normal lateral inhibitory connections. As an alternative, a linear network with inhibitory interneurons (Fig. 4.7(b)) can be used. After an initial transient, this network settles to

$$\mathbf{y} = \mathbf{x} - \mathbf{V}\mathbf{z} \quad \text{and} \quad \mathbf{z} = \mathbf{V}^T \mathbf{y} \quad (4.29)$$

i.e.

$$\mathbf{y} = (\mathbf{I} + \mathbf{V}\mathbf{V}^T)^{-1} \mathbf{x} \quad (4.30)$$

where v_{ij} is now the weight of the excitatory (positive) connection from y_i to z_j , and also the weight of the inhibitory (negative) connection back from z_j to y_i .

Suppose that the weights in this network are updated according to the algorithm

$$\Delta v_{ij} = \eta(y_i z_j - (1/\lambda)v_{ij}) \quad (4.31)$$

which is a Hebbian (or anti-Hebbian) algorithm with weight decay, and is

$$\Delta \mathbf{V} = \eta(\mathbf{C}_Y - (1/\lambda)\mathbf{I}_M)\mathbf{V} \quad (4.32)$$

in vector notation. Then the algorithm will converge when $\mathbf{C}_Y = (1/\lambda)\mathbf{I}_M$, which is precisely what we need to maximise J . In fact, this algorithm will also monotonically increase J as it progresses.

This network suggests that inhibitory interneurons, which are found in many places in sensory systems, may be performing some sort of

decorrelation task. Not only does the condition of decorrelated equal variance output optimize information transmission for a given power cost, but it can be achieved by various biologically plausible Hebb-like algorithms.

4.4.3 Introducing more realistic constraints

A real visual system such as the retina has to deal with noise on both the incoming signal (due to photon shot noise) and noise on the output of the network (due to e.g. spiking shot noise). Under certain simplifying conditions (e.g. the output noise dominates) it is possible to develop networks with simple local algorithms which can optimize information with both noise sources [17].

In more realistic situations it is more difficult to develop learning algorithms, but information theory can still be applied with very interesting results. For example, Atick and Redlich [2] derived a theoretical prediction for the spatial characteristics of retinal filters as a function of background light levels. This theoretical prediction shows a remarkable agreement with human psychophysical contrast sensitivity measurements.

More recently, Dong and Atick [8] have extended this work to the temporal domain, including the effect of nonlinearities in the visual pathway. They suggest that the lateral geniculate nucleus (LGN), until recently considered to be simply a relay station between the retina and the visual cortex, in fact helps to optimize the representation of visual information in the temporal domain. Since the neurons in the LGN can only have a positive firing rate, some cells represent the positive (nonlagged) parts of the signal, while others represent the negative (lagged) parts.

4.5 I-Max: Maximising Mutual Information between Output Units

In a slightly different use of information theory to those outlined above, Becker and Hinton [7] suggested that the information *between* output units could be used as the objective function for an unsupervised learning technique (figure 4.8).

In a visual system, this scheme would attempt to extract higher-order features of the visual scene which are coherent over space (or time). For example, if two networks each produce a single output from two separate but neighbouring *patches* of a retina, the objective of their algorithm is to maximise the mutual information $I(Y_1, Y_2)$ between these two outputs. A steepest-ascent procedure can be used to find the maximum of this mutual information function, both for binary and real-valued units.

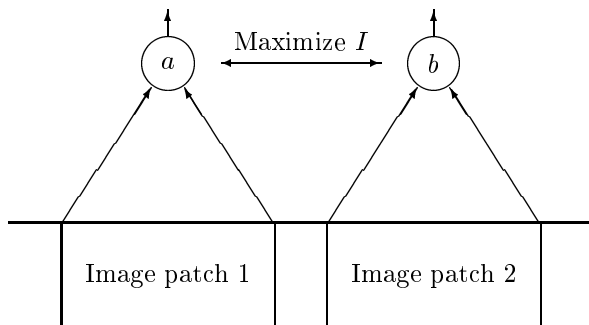


Figure 4.8. Two units with separate inputs, that maximize their mutual information.

One application of this principle is the extraction of depth from random-dot stereograms. Nearby patches in an image usually view objects of a similar depth, so if the mutual information between neighbouring patches is to be maximised, the outputs from both output units y_1 and y_2 should correspond to the information which is common between the patches, rather than that which is different. In other words the outputs should both learn to extract the common depth information rather than any other property of the random dot patterns.

For binary-valued units, with each unit similar to that used by the G-max scheme described above, the mutual information $I(Y_1, Y_2)$ between the two output units is

$$I(Y_1, Y_2) = H(Y_1) + H(Y_2) - H(Y_1, Y_2) \quad (4.33)$$

so if the probability distributions $P(y_1)$, $P(y_2)$ and $P(y_1, y_2)$ are measured, this mutual information can be calculated directly. Of course, it is sufficient to measure $P(y_1, y_2)$ only, since

$$P(y_1) = \sum_{y_2=0}^1 P(y_1, y_2)$$

and similarly for $P(y_2)$. The derivative of (4.33) can be taken with respect to the weights in the network for each different input pattern, so enabling the steepest-ascent procedure to be used.

For real-valued outputs it would be impossible to measure the entire probability distribution $P(Y_1, Y_2)$, so instead it is assumed that the two outputs have a Gaussian probability distribution, and that one of the

outputs is a noisy version of the other, with independent additive Gaussian noise. In this case, the information $I(Y_1, Y_2)$ between the two outputs can be calculated from the variances of one of the outputs (the ‘signal’) and the variance of the difference between the outputs (the ‘noise’) as

$$I(Y_1, Y_2) = \frac{1}{2} \log \frac{\sigma_{Y_1}^2}{\sigma_{Y_1 - Y_2}^2} \quad (4.34)$$

where $\sigma_{Y_1}^2$ is the variance of the output of the first unit, and $\sigma_{Y_1 - Y_2}^2$ is the variance of the difference between the two outputs.

If we accumulate the mean and variance of both Y_1 and $Y_1 - Y_2$, it is possible to find the derivative of (4.34) for each input pattern, with respect to each weight value. Thus the weights in the network can be updated in a steepest-ascent procedure to maximise $I(Y_1, Y_2)$, or at least the approximation to $I(Y_1, Y_2)$ given by (4.34).

Becker and Hinton found that unsupervised networks using this principle could learn to extract depth information from random-dot stereograms with either binary- or continuous-valued shifts, as appropriate for the type of outputs used, although in some cases it helped to force the units to share weight values, further enforcing the idea that the units should calculate the same function of the input. They generalised their scheme to allow networks with hidden layers, and also to allow multiple output units, with each unit maximising the mutual information between itself and a value predicted from its neighbouring units. This latter scheme allowed the system to discover an interpolation for curved surfaces.

4.6 Conclusions

In this paper we have introduced some of the ways that information theory is used with neural networks.

We have seen that noise, which is not often important in many neural networks, is of central importance when dealing with information theory. Noise limits the information which can be transmitted by a network. If the input has spherical Gaussian noise, information is optimized if we use a network which extracts the principal components of the input suppressing low-amplitude input components. If noise is on the output, a decorrelating (whitening) network is needed, boosting low-amplitude input components. In more realistic situations, where noise is present on both input and output, a trade-off between these two extremes emerges.

As an illustration of other possibilities, we have also seen how common information can be extracted from network outputs using the ‘I-max’ approach. Although the depth is only a relatively small amount

of the total information in a random-dot stereogram, maximizing the information between the outputs of networks connected to neighbouring patches extracts the depth because no other information is common to both patches.

Although we have concentrated on unsupervised learning in this paper, information theory can also give us insight into supervised learning. For example, the graded outputs for a multi-layer perceptron contain more information about the likely output than the identity of the ‘best’ output alone. If the output from this supervised network is to be used in some further processing stage, we could keep more information by, for example, keeping some m best outputs instead of just the single best output [15].

Information theory has proved to be a useful tool in the neural networks armoury. In particular, it has given us a ‘driving force’ for unsupervised neural network learning algorithms. The arguments that biological information processing systems are optimizing information are backed up by good results from early vision. These suggest that in the future, information theory could offer us a way to investigate how perception works *above* the level of individual neurons, by considering what happens to the *information*, not just to individual signals.

References

- [1] J. J. Atick and A. N. Redlich. Towards a theory of early visual processing. *Neural Computation*, 2:308–320, 1990.
- [2] J. J. Atick and A. N. Redlich. What does the retina know about natural scenes? *Neural Computation*, 4:196–210, 1992.
- [3] J. J. Atick and A. N. Redlich. Convergent algorithm for sensory receptive field development. *Neural Computation*, 5:45–60, 1993.
- [4] F. Attneave. Some informational aspects of visual perception. *Psychological Review*, 61:183–193, 1954.
- [5] H. B. Barlow. Three points about lateral inhibition. In W. Rosenblith, editor, *Sensory Communication*, pages 782–786. MIT Press, 1961.
- [6] H. B. Barlow and P. Földiák. Adaptation and decorrelation in the cortex. In *The Computing Neuron*, pages 54–72. Addison-Wesley, Wokingham, England, 1989.
- [7] S. Becker and G. E. Hinton. Spatial coherence as an internal teacher for a neural network. Technical Report CRG-TR-89-7, Department of Computer Science, University of Toronto, December 1989.
- [8] D. W. Dong and J. J. Atick. Temporal decorrelation: a theory of lagged and nonlagged responses in the lateral geniculate nucleus. *Network: Computation in Neural Systems*, 6:159–179, 1995.

- [9] P. Földiák. Adaptive network for optimal linear feature extraction. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN-89*, pages 401–405, New York, June 1989. IEEE Press.
- [10] D. O. Hebb. *The Organization of Behavior*. Wiley, New York, 1949.
- [11] S. W. Kuffler, J. G. Nicholls, and A. R. Martin. *From Neuron to Brain: A Cellular Approach to the Function of the Nervous System*. Sinauer Associates Inc., Sunderland, MA, second edition, 1984.
- [12] R. Linsker. Self-organization in a perceptual network. *IEEE Computer*, 21(3):105–117, March 1988.
- [13] E. Oja. A simplified neuron model as a principal component analyser. *Journal of Mathematical Biology*, 15:267–273, 1982.
- [14] E. Oja and J. Karhunen. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications*, 106:69–84, 1985.
- [15] M. D. Plumbley. On information theory and unsupervised neural networks. Technical Report CUED/F-INFENG/TR.78, Cambridge University Engineering Department, UK, 1991.
- [16] M. D. Plumbley. Efficient information transfer and anti-Hebbian neural networks. *Neural Networks*, 6:823–833, 1993.
- [17] M. D. Plumbley. A Hebbian/anti-Hebbian network which optimizes information capacity by orthonormalizing the principal subspace. In *Proceedings of the IEE Artificial Neural Networks Conference, ANN-93, Brighton, UK*, pages 86–90, 1993.
- [18] M. D. Plumbley. Lyapunov functions for convergence of principal component algorithms. *Neural Networks*, 8:11–23, 1995.
- [19] M. D. Plumbley and F. Fallside. An information-theoretic approach to unsupervised connectionist models. In D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 239–245. Morgan-Kaufmann, San Mateo, CA., 1988.
- [20] T. D. Sanger. An optimality principle for unsupervised learning. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 1*, pages 11–19. Morgan Kaufmann, San Mateo, CA, 1989.
- [21] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.
- [22] C. E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37:10–21, 1949.
- [23] C. von der Malsburg. Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85–100, 1973.
- [24] S. Watanabe. *Pattern Recognition: Human and Mechanical*. John Wiley & Sons, New York, 1985.
- [25] R. J. Williams. Feature discovery through error-correction learning. ICS Report 8501, University of California, San Diego, 1985.