Communicated by Ralph Linsker

Redundancy Reduction as a Strategy for Unsupervised Learning

A. Norman Redlich
The Rockefeller University, 1230 York Ave., New York, NY 10021 USA

A redundancy reduction strategy, which can be applied in stages, is proposed as a way to learn as efficiently as possible the statistical properties of an ensemble of sensory messages. The method works best for inputs consisting of strongly correlated groups, that is *features*, with weaker statistical dependence between different features. This is the case for localized objects in an image or for words in a text. A local feature measure determining how much a single feature reduces the total redundancy is derived which turns out to depend only on the probability of the feature and of its components, but not on the statistical properties of any other features. The locality of this measure makes it ideal as the basis for a "neural" implementation of redundancy reduction, and an example of a very simple non-Hebbian algorithm is given. The effect of noise on learning redundancy is also discussed.

1 Introduction

Given sensory messages, for example, the visual images available at the photoreceptors, animals must identify those objects or scenes that have some value to them. This problem, however, can be very tricky since the image data (e.g., photoreceptor signals) may underdetermine the scene data (e.g., surface reflectances) needed to find and identify objects (Kersten 1990). In the case of very primitive organisms crude special purpose filters may suffice, such as the "fly detector" in frogs. But for more general object detection and for the reconstruction of physical scenes from noisy image data, some additional clues or constraints are needed. One type of clue is knowledge of the *statistical properties* of scenes and images (Attneave 1954, Barlow 1961, 1989). Such information can be used to recover physical scene data from noisy image data, as shown for example by Geman and Geman (1984). Barlow (1989) has also argued that such information is necessary for object recognition, since it allows objects to be discriminated from irrelevant background data. Also, since objects are encoded *redundantly* in sensory messages, knowing this redundancy can aid in their recognition.

But how can an organism go about *learning* the statistical properties of sensory messages? And second, what is the most efficient way of *stor-*

 © 1993 Massachusetts Institute of Technology

ing this statistical knowledge? The enormity of these problems becomes obvious when one considers just how many numbers in principle must be learned and stored. In vision this amounts to storing the probability of every possible set of pixel values in both space and time. For a conservative estimate of this number for humans, assume one million cones sampled in space only—temporal sampling would add considerably to this. Then assume a grey scale of roughly 100, which is less than the number of contrast units that can be discriminated in bright light—and ignores luminance data. This gives $100^{1,000,000}$ possible images whose probabilities could not possibly be stored as $100^{1,000,000}$ numbers in the brain, which has no more than 10^{16} synapses. However, there are two very important properties of images which allow this number to be decreased enormously. The first and most obvious is noise: most images differ from each other only by noise or by symmetries, so there is no need to learn and store their individual probabilities.

The second simplifying property is that sensory message probabilities can often be derived from a far smaller set of numbers. This is the case when the set of probabilities $P(I)$ for the images $I = \{l_1, l_2, l_3, \ldots l_n\}$, with pixel values l_n, can be factorized into a far smaller set of statistically independent probabilities for the subimages $\{I_1, I_2, I_3, \ldots I_m\}$ as $P(I) = P(I_1)P(I_2)P(I_3)\ldots P(I_m)$. Thus, as Barlow (1989) has emphasized, the most efficient way to store the probabilities $P(I)$ would be to find a transformation, a *factorial code*, from the pixel representation l_n to the statistically independent representation I_m with smallest m. It can be demonstrated (Atick and Redlich 1990a) that this explains one purpose of the retinal transfer function, which approximately removes (second-order) statistical dependence from the optic nerve outputs $\{I_1, I_2, I_3, \ldots I_m\}$.

Finding a transformation to a factorial representation is an unsupervised learning problem that typically requires many learning stages. At each stage I assume that only the local probabilities $P(l_n)$ are measured, but as statistical independence is "increased", products of these give better approximations to the joint probabilities $P(I)$. To quantify just how statistically independent a representation is at each stage, it is necessary to define a *global* learning measure $\mathcal{L}$, which should be a function only of the local probabilities $P(l_n)$ (global denotes a property of the entire representation at each stage). Such a measure is defined here based on the *redundancy*,[1] a quantity that is minimal only when the code is factorial.

Learning a redundancy reducing transformation at each stage can be very difficult and may depend on the nature of the redundancy at that stage. In the retina, the greatest source of redundancy is due to

[1] I use "redundancy reduction" to refer to statistical dependence between pixels. This is not strictly speaking the only source of redundancy, which also can come from the uneven probability distribution of grayscale values. Nevertheless, I use the term "redundancy reduction" because it implies an information preserving transformation (unlike, e.g., "entropy reduction") and also because the word "redundancy" has an intuitive appeal. The precise meaning of redundancy reduction here is a transformation which increases the learning measure $\mathcal{L}$, to be defined in Section 2.

multiscale second-order correlations between pixels—corresponding to scenes being very smooth over large spatial and temporal regions—and this redundancy can be removed easily through a *linear* filter (Atick and Redlich 1991). But this is the exception, since in general quite complicated nonlinear coding is required (for some progress see, e.g., Barlow and Foldiak 1989; Redlich 1992). However, there is one common type of nonlinear redundancy reduction that is relatively straightforward to learn. This is the redundancy in images coming from strong correlations within sharply delineated *features* which are in turn weakly correlated with each other (the features can be spatially extended as long as they decouple from other parts of the image).

The procedure for factorizing in this case is to look first for the subfeatures that are most tightly bound, and therefore are responsible for the most redundancy. These may then be pieced together in stages, until eventually a statistically independent set is found. What makes this much simpler than I expected is the existence of a completely *local* measure of how much an individual feature (or subfeature) contributes to the global redundancy. By local I mean that this measure is only a function of the probabilities of the feature and its components, but *not* of the probabilities of any other feature or components.

The locality of this feature measure also allows simple implementation of redundancy reduction through unsupervised "neural" learning algorithms. One such non-Hebbian algorithm will be discussed here, and compared to some other unsupervised algorithms (von der Malsburg 1973; Bienenstock *et al.* 1982; Hinton and Pearlmutter 1986). The closest connection is with Hinton and Pearlmutter's algorithm because their single-unit feature measure is mathematically related to the one here, though this is manifest only in a particular approximation. This connection is not surprising since (see, e.g., Hinton and Sejnowski 1983) their aim was also to learn statistical regularities. Some of the major distinctions between this work and theirs are the focus here on efficiency of storage and learning (on statistical independence) and also the insistence here on transformations which preserve information (see Section 6).

To demonstrate the power of the present approach, I apply it to strip the redundancy from English text—to learn the text's statistical properties. This example is used because we all know a fairly good solution to the problem: transform from the letter representation to the word representation. Of course to make the problem sufficiently difficult all clues to the solution, such as spaces between words, punctuation, and capitalization, are first eliminated from the text. The algorithm eventually segments the text just as desired into words and tightly bound groups of words.

Although it is not the main purpose of this paper, I shall also indicate how useful the algorithm can be for recovering messages from noisy signals. This works best when the useful information is coded redundantly while the noise is random. It then turns out that the algorithm used here

finds only the useful portion of the input, and this will be demonstrated using noisy text.

Finally, I should emphasize that my aim is *not* to find redundancy in language or to claim that words are learned or stored in the brain as found here. Instead, my ultimate motivation is to find an environmentally driven, self-organizing principle for the processing of visual images (and other sensory signals) to facilitate object or pattern identification (see Redlich 1992). So by "words" here I always wish to imply visual features, with letter positions in a text corresponding to pixel locations in an image and particular letters corresponding to image grayscale (or color) values. The next step of applying the algorithms derived here to visual images will appear in future papers.

2 Global Learning Measure $\mathcal{L}$

Taking the English text example, the input consists of an undifferentiated stream of "pixel" values $L = \{a, b, c, \ldots\}$ (as in Fig. 1a) and the goal is to learn the probability functions $P(I) = P(l_1, l_2, l_3, \ldots l_n)$, with the subscript n denoting the position of letter l_n in the text. In practice, the aim is to learn $P(I)$ for string length n roughly equal to the correlation length of the system. But even for n as small as 12 this in principle requires storing and updating as many as 26^{12} numbers.

To find $P(I)$ more efficiently, at each stage letters will be grouped together into "words," which at first will be only pieces of real English words. Then at successive stages the new set of words $W = \{w_1, w_2, w_3, \ldots w_m\}$ will be built by combining the previous "words" into larger ones, among which will be *real* English words and also tightly correlated groups of real words (from now on quotes around words are dropped). At the very first stage $P(I) = P(l_1, l_2, l_3, \ldots l_n)$ is very poorly approximated by the product of letter probabilities $P(l_1)P(l_2)P(l_3)\ldots P(l_n)$, but as the redundancy is reduced, products of word probabilities $P(w_1)P(w_2)P(w_3)\ldots P(w_m)$ give better and better approximations to $P(I)$.

To quantitatively measure how well $P(I)$ is known at each stage, we can use information theory (see also Barlow 1961) to define a global *learning measure*

$$\mathcal{L} = 1 - \frac{(H_W/\bar{s})}{H_L} \tag{2.1}$$

where H_L is the entropy in the original letter code,

$$H_L = -\sum_{l \in L} P_l \log(P_l) \tag{2.2}$$

and $H_W/\bar{s}$ is the word entropy per letter at a particular stage in learning:

$$\begin{aligned} H_W &= -\sum_{w \in W} P_w \log(P_w) \\ \bar{s} &= \sum P_w s_w \end{aligned} \tag{2.3}$$

with $\bar{s}$ the average of the word lengths s_w.

---

a l i c e w a s b e g i n n i n g t o g e t v e r y t i r e d o f s i t t i n g b y h e r s i s t e r o n
t h e b a n k a n d o f h a v i n g n o t h i n g t o d o o n c e o r t w i c e s h e h a d p e e p
e d i n t o t h e b o o k h e r s i s t e r w a s r e a d i n g b u t i t h a d n o p i c t u r e s o r
c o n v e r s a t i o n s i n i t a n d w h a t i s t h e u s e o f a b o o k t h o u g h t a l i c e w
i t h o u t p i c t u r e s o r c o n v e r s a t i o n s s o s h e w a s c o n s i d e r i n g i n h e
r o w n m i n d a s w e l l a s s h e c o u l d f o r t h e h o t d a y m a d e h e r f e e l v e r y
s l e e p y a n d s t u p i d w h e t h e r t h e p l e a s u r e o f m a k i n g a d a i s y c h a i
n w o u l d b e w o r t h t h e t r o u b l e o f g e t t i n g u p a n d p i c k i n g t h e d a i s i
e s w h e n s u d d e n l y a w h i t e r a b b i t w i t h p i n k e y e s r a n c l o s e b y h e r
t h e r e w a s n o t h i n g s o v e r y r e m a r k a b l e i n t h a t n o r d i d a l i c e t h i
n k i t s o v e r y m u c h o u t o f t h e w a y t o h e a r t h e r a b b i t s a y t o i t s e l f o
h d e a r o h d e a r

(a)

alice was b e g in n ing t o g e t very t i r e d o f s i t t ing b y her s i s t e r o n the b a n
k and o f h a v ing n o th ing t o d o o n c e o r t w i c e she h a d p e e p e d in t o the b o
o k her s i s t e r was r e a d ing but i th a d n o p i c t u r e s o r c o n v e r s a t i o n s in i
t and w h a t i s the u s e o f a b o o k th ough t alice with ou t p i c t u r e s o r c o n v e r
s a t i o n s s o she was c o n s i d e r ing in her o w n m in d a s w e l l a s she c ould for
the h o t d a y m a d e her f e e l very s l e e p y and s t u p i d w he the r the p l e a s u r
e o f m a k ing a d a i s y c h a in w ould b e w o r th the t r ou b l e o f g e t t ing u p
and p i c k ing the d a i s i e s w h e n s u d d e n l y a w h i t e r a b b i t with p in k e y e
s r a n c l o s e b y her the r e was n o th ing s o very r e m a r k a b l e in that n o r d i
dalic e th in k i t s o very m u c h ou t o f the w a y t o he a r the r a b b i t s a y t o i t s e
l f o h d e a r o h d e a r

(b)

alice was beg in n ing to g e t very t i r ed of s it t ing b y her s i s t e r on the b a n k
and of ha v ing n o thing to d o on c e o r t w i c e shehad p e e p ed in to the b ook her
s i s t e r was r e a d ing but i th a d n o p i c t u r e s o r c on ver s a t i on s in it and
what i s the u s e of a b ook thought alice with ou t p i c t u r e s o r c on ver s a t i on s
s o she was c on s i d e r ing in her own m in d a s w e l l a s shecould for the h o t d a y
m a d e her f e e l very s l e e p y and s t u p i d w he ther the p l e a s u r e of m a k ing
a d a i s y ch a in w ould be w o r th the t r ou b l e of g e t t ing u p and p i c k ing the d
a i s i e s when s u d d e n ly a whi t e r a b b it with p in k e y e s r a n c l o s e b y her
ther e was n o thing s o very r e m a r k a b l e in that n o r d i dalic e think it s o very
m uch ou t ofthe way to he a r ther a b b it s a y to it s elf o h d e a r o h d e a r

(c)

Figure 1: A small sample of the text with all clues to the redundancy removed. In (a) single letters are treated as words, indicated by the spaces between them, and the entropy is $H_L = 4.16$ bits. As the redundancy is reduced letters are combined into words, indicated by removing spaces between them. Only some of the redundancy reduction stages are shown in (b)–(f), with the entropy per letter reduced to $H_W/\bar{s} = 3.75$, 3.46, 2.84, 2.51, and 2.35 bits, respectively (the real word entropy per letter is $H_W/\bar{s} = 2.17$ bits). *Continued.*

alice was beg in n ing toge t very t ir ed of s it t ing by her s is ter on the b an k and of having nothing to do on ce ort w ice shehad p e e p ed intothe b ook her s is ter was read ing but i thad no p i c tur e s or conversation s in it and what is the us e of a b ook thoughtalice without p i c tur e s or conversation s s o shewas con side r ing in her own m ind as well asshecould for the h o t day made her fee l very s l e e p y and s t up id w he ther the pleas ure of mak ing ad a is y ch a in would be wor th the t r ou ble of getting up and p i ck ing the d a is i e s when suddenly a whiterabbit with p in key e s r an close by her therewas nothing s o very re mark a ble in that no r d i dalic e think it s o very much outofthe way to hear therabbit say to it self ohdear ohdear

(d)

alice was beginn ing toget verytiredof s it t ing by her s is ter on the bank and of having nothing to do on ce ortwice shehad peeped intothe book her s is ter was read ing but ithad no picture s or conversation s in it and what is the us e of a book thoughtalice without picture s or conversation s so shewas consider ing in her own mind aswell asshecould for the hot day made her fee l very sleep y and stupid whether the pleas ure of making ad a is y ch a in wouldbe wor th the t rouble of getting up and p ick ing the d a is i e s when suddenly a whiterabbit with p in key e s r an close by her therewas nothing so very remark a ble in that no r d i dalic e think it so very much outofthe way to hear therabbit say to it self ohdear ohdear

(e)

alice was beginning toget verytiredof sitting by her s is ter onthebank and of having *nothingtodo onceortwice shehad peeped intothe* book her s is ter was read ing but ithad no picturesor conversation s in it and what is the us e of a book thoughtalice without picturesor conversation s so shewas consider ing in her own mind aswell asshecould for the hot day made her feelvery sleepyand stupid whether the pleas ure of making ad a is y ch a in wouldbe wor th the t rouble of getting up and p ick ing the d a is i e s when suddenly a whiterabbit with p in key e s r an close by her therewas nothing so very remark able in that no r d i dalic e think it so very much outofthe way to hear therabbit say toitself ohdear ohdear

(f)

Figure 1: Continued.

Initially, the set of words W is also the set of letters L so $H_W/\bar{s} = H_L$ and $\mathcal{L} = 0$, indicating that no learning has occurred. At the other extreme the word code is factorial for which[2] $H_W/\bar{s} = H$, where H is the total entropy per letter of the text:

$$H = \lim_{n\to\infty} \left\{ \sum_{I=\{l_1,l_2,l_3,\ldots l_n\}} -P(I)\log(P(I)) \right\} /n \tag{2.4}$$

[It is well known (Shannon and Weaver 1949) that $H_W/\bar{s} \geq H$ with equality only when the words w in W become completely independent.] So

[2]This is true because the word code is reversible, so H is invariant; for another type of reversible code see Redlich (1992).

the learning measure $\mathcal{L}$ starts out equal to zero and grows as redundancy is reduced until it approaches its maximum value

$$\mathcal{L} \rightarrow \mathcal{R}_c = 1 - H/H_L \tag{2.5}$$

where $\mathcal{R}_c$ is the total *redundancy* in the text due to correlations between letters. If there are no correlations between letters then $H = H_L$ and $\mathcal{R}_c = 0$.

It is important to note that although $\mathcal{L}$ is bounded from above by $\mathcal{R}_c$ ($H_W/\bar{s}$ is bounded from below by H), it *can* go negative, so the system can in effect unlearn or increase redundancy. This happens when words (or letters) at one stage which are already independent of each other are mistakenly combined into new words.

3 Local Feature Measure $\mathcal{F}$

Now that we have a global learning measure $\mathcal{L}$, how do we go about finding the word/letter combinations $W \rightarrow W'$, which increase $\mathcal{L}$? For this purpose it is useful to have a *local* measure of how much an individual new word or *feature* increases $\mathcal{L}$. Such a local feature measure $\mathcal{F}$ can be derived directly from $\mathcal{L}$ by calculating the change in $\mathcal{L}$ caused by including in $W \rightarrow W'$ a single new feature. Actually, since increasing $\mathcal{L}$ corresponds to decreasing $H_W/\bar{s}$, we need to calculate the change in $H_W/\bar{s}$.

For extra clarity, let us first calculate the change in $H_W/\bar{s}$ when only two words in W are combined to form a new word. Assume for simplicity that the current word set W still contains many single letters, including the letters "*i*" and "*n*." Let us see, as an example, how combining these letters into the particular word $w =$ "*in*" changes $H_W/\bar{s}$ in 2.3. Following $W \rightarrow W'$

$$\begin{aligned} H'_w &= -P'_{in}\log(P'_{in}) - \sum P'_w \log(P'_w) \\ \bar{s}' &= P'_{in}s'_{in} + \sum P'_w s'_w \end{aligned} \tag{3.1}$$

where P_{in}, P_i, and P_n denote the probabilities of the example word "*in*," and of the letters "*i*" and "*n*." Also the "*in*" terms have been separated out, so the sum $\sum$ still runs over the old set W (assuming "*i*" and/or "*n*" still exist as independent elements in the set W').

To calculate the change $H'_w/\bar{s}' - H_w/\bar{s}$, the new probabilities P' must be expressed in terms of the old probabilities P. This is easily accomplished using $P_w = N_w/N$, where $N_w =$ number of times word w appears, and $N =$ total word count for the text (later N can be taken to infinity). After combining "*i*" and "*n*" into "*in*," the number $N \rightarrow N' = N - N_{in}$, since every time the word "*in*" occurs it is counted as one word in W', but was

counted as two words in W. Likewise, $N_i \to N_i - N_{in}$, and $N_n \to N_n - N_{in}$. Therefore,

$$\begin{aligned} P_{in} \to P'_{in} &= N_{in}/(N - N_{in}) = P_{in}/(1 - P_{in}) \\ P_{i,n} \to P'_{i,n} &= (N_{i,n} - N_{in})/(N - N_{in}) = (P_{i,n} - P_{in})/(1 - P_{in}) \\ P_w \to P'_w &= N_w/(N - N_{in}) = P_w/(1 - P_{in}), \text{ for } w \neq \text{"}i\text{,"} \text{ "}n\text{"} \end{aligned} \tag{3.2}$$

Substituting these P' into 3.1 gives

$$\begin{aligned} H'_w &= \frac{-1}{1 - P_{in}} \left\{ P_{in} \log\left(\frac{P_{in}}{1 - P_{in}}\right) + \sum_{w=i,n} (P_w - P_{in}) \log\left(\frac{P_w - P_{in}}{1 - P_{in}}\right) \right. \\ &\quad \left. + \sum_{w \neq i,n} P_w \log\left(\frac{P_w}{1 - P_{in}}\right) \right\} \\ \bar{s}' &= \frac{-1}{(1 - P_{in})} \left\{ P_{in} s_{in} + \sum_{w=i,n} (P_w - P_{in}) s_w + \sum_{w \neq i,n} P_w s_w \right\} \\ &= \bar{s}/(1 - P_{in}) \end{aligned} \tag{3.3}$$

so that

$$(H_W/\bar{s})' = H_W/\bar{s} - \mathcal{F}/\bar{s} \tag{3.4}$$

which defines the feature measure $\mathcal{F}$

$$\begin{aligned} \mathcal{F} &= \mathcal{F}(P_{in}, P_i, P_n) \\ &= \left\{ P_{in} \log\left(\frac{P_{in}}{P_i P_n}\right) + \sum_{w=i,n} (P_w - P_{in}) \log\left(\frac{1 - P_{in}}{P_w}\right) - (1 - P_{in}) \log(1 - P_{in}) \right\} \end{aligned} \tag{3.5}$$

The original average word length $\bar{s}$ has not been included in the definition of $\mathcal{F}$ because it is the same for all new features built out of W. As promised, this feature measure depends only on the *local* data P_{in}, P_i, and P_n.

The local measure $\mathcal{F}$ can also be derived in general for new words of any length. We need to take into account the number m of old words making up the new word, as well as the number of times, m_w, each old word w appears. For example if the new word "trees" is built out of the old words "*tr*," "*e*," and "*s*" in W, then $m = 4$, while $m_{tr} = 1$, $m_e = 2$, $m_s = 1$ giving $\sum m_w = m$. The new probabilities P' can then be derived from P_w and m_w using counting arguments only slightly more complicated than before. Thus, denoting the new word by f, for feature,

$N \rightarrow N' = N - (m-1)N_f$, while $N_w \rightarrow N_w - m_w N_f$ for w in the set W_f of old words in f, and $N_w \rightarrow N_w$ otherwise. With these adjustments, the general feature measure defined by 3.4 is

$$\begin{aligned}\mathcal{F} &= \mathcal{F}(P_f, P_w\ w \in W_f) \\ &= \left\{ P_f \log\left(\frac{P_f}{\prod_{w\in W_f} P_w^{m_w}}\right) + \sum_{w\in W} (P_w - m_w P_f)\log(1 - m_w P_f/P_w) \right. \\ &\quad \left. -[1-(m-1)P_f]\log[1-(m-1)P_f] \right\} \qquad (3.6)\end{aligned}$$

This reduces to 3.5 in the special case $f =$ "*in*," $W_f = \{$"*i*," "*n*"$\}$, $m = 2$, $m_i = 1$, $m_n = 1$.

To gain some intuition into just what statistical properties are measured by $\mathcal{F}$ it is useful to approximate $\mathcal{F}$ for the case $P_f << P_w$, for all $w \in W_f$. In the case of English text, this is a good approximation in the first stages of redundancy reduction. The approximate $\mathcal{F}$ is

$$\mathcal{F} \cong P_f \left\{ \log\left(\frac{P_f}{\prod_{w\in W_f} P_w^{m_w}}\right) - 1 \right\} \qquad (3.7)$$

using log base e. In this form, $\mathcal{F}$ bears its closest resemblance to the single-unit G-maximization measure of Hinton and Pearlmutter (1986). This is because the first term in 3.7, that is, neglecting $-P_f$, is like a single term in the Kullback information when the original probability estimate is statistical independence of inputs. The actual Kullback information requires summing such terms over all features f.

For the feature f to significantly reduce redundancy, $\mathcal{F}$ needs to be strongly positive and for this, we see from 3.7 that the feature must have *two* statistical properties: First, the term in parentheses must be large and positive, which requires $P_f >> \prod P_w$. This term plus one is the *mutual information* (Shannon and Weaver 1949), which measures how strongly the components that make up the feature,[3] are correlated. A good example in English is "*qu*" which has high mutual information because "*q*" always predicts "*u*" so the "*u*" following "*q*" is redundant.

The second requirement is that the feature be relatively frequent since P_f multiplies the mutual information. Otherwise, the feature could be highly self-correlated, but not common enough to significantly reduce the global redundancy. This is very important, since the mutual infor-

[3]In physics language, the feature is analogous to a bound state like an atom built out of protons and electrons. The mutual information is then proportional to the difference between the bound state (feature) energy and the sum of the energies of its components. This is the amount of energy that is gained by building the bound state (atom).

mation alone tends to favor very rare features composed of very rare elements. On the other hand, large P_f alone is a dangerous criterion since there are many common features with small or even negative mutual information. Including these in the new set W' actually *increases* the redundancy, since it effectively creates a correlated structure out of already statistically independent elements. In English text an example of a redundancy increasing feature is *"tte,"* built out of *"t"* and *"e."*

4 Experimental Results

$\mathcal{F}$ can be applied to devise a redundancy reduction algorithm for English text. One simple strategy would be to find at each stage the single new word which has the largest $\mathcal{F}$, and thus find W' from W. However, in practice it turns out that a far more time efficient approach is to find the *set* of new words with largest $\mathcal{F}$, say 10 to 100 new words at each step. Another computational efficiency is gained by limiting the number m of component words to some small number, such as three or four. It turns out that for English text—likely also for many other ensembles—using only up to third-order correlations at each stage ($m = 2, 3$) is sufficient, since larger words are most often composed of redundant subwords.

To experimentally test how well this works I applied it to learning about 25 pages of a well known children's book (Carroll 1865), chosen for its moderately sized vocabulary of roughly 1700 (real English) words. After eliminating all punctuation, capitals, and word spaces, the excerpt contained approximately 48,000 characters. The letter entropy was found to be $H_L = 4.16$ bits, while the entropy per letter for real English words is 2.17 bits. Figure 1a shows a small piece of the text after it was stripped of any redundancy clues. Spaces are used between letters to indicate that they are being treated here as separate "words." Figure 1b–f then show the text sample in various stages of redundancy reduction. At each stage, when new words are built the spaces between their component words are eliminated. Figure 1 shows the results of using only second- and third-order joint probabilities at each step to find roughly 10 to 100 new words per stage. About 20 such stages were required to get the redundancy down to the $H_W = 2.35$ bits of Figure 1e (only 5 of the 20 stages are actually shown in the figure) which is close to the real word entropy. Even computing all second- and third-order joint probabilities, these results represent only a few hours computation on a Macintosh computer. But the computation time and array storage needed can be reduced even further by calculating the joint probabilities only for a sample of possible new words, as will be discussed in the next section.

Figure 2 shows the improvement possible using up to fourth-order probabilities; only the last stage is shown in the figure. Since there is only a small improvement over the third-order result, this demonstrates that fourth-order is not absolutely necessary.

alice was beginning toget verytiredof sitting by hersister ontheban k and of having nothingtodo onceortwice shehad peeped intothe book hersister was read ing but ithad no pictures or conversation s in it and what is theuseof a book thoughtalice without pictures or conversation s so shewas consider ing in her ow n mind aswell asshecould for the hot day made her feel very sleepy and stupid whether the plea sure of making ad a is y ch a in wouldbe wor th the trouble of gettingupand p i c king the d a is ies when suddenly a whiterabbit with p in k eyes r an close by her therewas nothing so very remark able in that n or did alice think it so very much outoftheway to hear therabbit saytoitself ohdear ohdear

Figure 2: The same sample of text, but using up to fourth-order correlation per stage instead of the third-order limit in Figure 1. Only the last stage is shown. It has $H_W/\bar{s} = 2.28$ bits.

Reviewing the results in Figure 1, one may note that some real English words, such as "daisies," are not found, but this is due to the relatively small sample of English text used. In fact, the word "daisies" appears in the text only once so it would have been an error for it to qualify as a redundant feature. However, the algorithm is superbly sensitive to redundant words which appear in the text as few as two or three times. Another thing to observe is that many groups of real words are combined into single features. Some of this reflects actual redundancy in English, for example "ofthe" is likely a truly redundant combination in English, but many of these, such as "whiterabbit" are only redundant for this sample text. Such real word groupings would have far lower redundancy (lower $\mathcal{F}$) in a much larger sample text which includes many different subjects and writing styles.

The most significant success of the redundancy reduction algorithm is the segmentation of the text, which is almost always broken at the boundary between real words. This efficient segmentation corresponds to finding a cover W' (Fig. 1e) of the entire sample with a small number of words—less than the number of real words. This is close to the smallest number of (approximately) statistically independent words. Such efficient segmentation would not have been found using an algorithm that chooses only high probability words.

5 Neural Implementation

In a "neural" implementation of the algorithm used in Section 4, neurons, or dendrites, calculate the *local* data P_f and P_w. Actually, only P_f needs to be calculated since the P_w are computed by the previous stage and may be encoded in the neural output strengths. Finding $\mathcal{F}(P_f, P_w)$ still requires some computation, but (especially in 3.7) this reduces essentially

to computing logarithms.[4] The real problem is not how to calculate P_f or $\mathcal{F}(P_f)$, but how to search the space of possible features for those with largest $\mathcal{F}$. One option is to convert this search to one over a set of (continuous) synaptic weights and then apply gradient descent to maximize $\mathcal{F}$. This is the technique used by Hinton and Pearlmutter (1986) to maximize the Kullback information. Though its application to $\mathcal{F}$ is somewhat different, I believe it might work, although I have not attempted it. Instead, I wish to explore here a more direct approach which avoids the convergence problems often associated with gradient descent.

The simplest and most direct approach would be to exhaustively calculate P_f for *all* features of size $\leq m$. Of course m small enough to make this computationally feasible might be too small to discover the redundancy. But, there really is no need for an exhaustive search, since a prerequisite for large $\mathcal{F}$ is large P_f, and a more limited sampling will usually find these common features. Then only those common features with sufficiently large $\mathcal{F}$ need be kept. I now use this to develop a temporal search algorithm.

Suppose first that there are a fixed number (smaller than needed for an exhaustive search) of feature neurons at each learning stage, which can be in one of two states, *occupied* or *free*. Occupied neurons respond to one feature, and their job is to quickly calculate a good approximation for $\mathcal{F}$. As soon as the occupied neuron discovers that $\mathcal{F}$ is below some constant threshold $\mathcal{F}^*$, it becomes free and is available to test another feature. The neurons are mutually inhibiting so no two neurons can be occupied by the same feature. Also there is some ordering to decide which free neuron takes the next possible feature.

To approximate $\mathcal{F}$, a neuron only needs an approximation for P_f since the P_w were calculated by the previous stage. How big P_f needs to be for $\mathcal{F}(P_f, P_w) > \mathcal{F}^*$ depends on the probabilities of the input elements P_w that make up the feature. In effect, the feature neuron uses a feature-dependent threshold $\Delta(P_w)$ for P_f. (If the criterion were simple frequency of the feature, on the other hand, one would use a fixed threshold Δ for P_f.) Features that are built out of infrequent inputs w have lower threshold for P_f, as can be seen most easily in 3.7.

The final ingredient is an approximation for $P_f(t)$ at time t, where $t = 0$ is the time when the neuron first picks up its feature. For this, I make a very simple choice: If the feature has occurred only once at time $t = 0$, then for $t > 0$ approximate $P_f(t) = 1/t$; if the feature occurs a second time at $t = T_1$ use for $t > T_1$, $P_f(t) = 2/t$; and if the feature has

[4]It should be noted that $-\log(P)$ has a very nice interpretation as the information in or *improbability* of the signal. If neurons have output strengths proportional to the information they carry, then the mutual information, one of the ingredients needed for $\mathcal{F}$, can be calculated through simple addition of neuronal outputs. This was suggested by Uttley (1979) as one of the attractions of using the mutual information to build a conditional probability computer (Singh 1966). Also, the idea that neurons signal improbability has been proposed by Barlow (1989), and there is evidence for this in the retina.

occurred $n + 1$ times use $P_f(t) = n/t$, which eventually approaches the true P_f for large n.

If at any time $P_f(t)$ drops below the threshold $\Delta(P_w)$, that is, $\mathcal{F}(t)$ drops below $\mathcal{F}^*$, then the occupied neuron is freed to search for other features. Of course, since for small t $P_f(t)$ may be a poor approximation, good features will be dropped occasionally, but these are likely to be picked back up again since they must be relatively frequent. On the other hand, the longer a neuron is occupied by a feature, the better the approximate $P_f(t)$ becomes and the less susceptible to such errors. In fact, I have simulated this algorithm for the beginning stages of learning for the sample text used in Section 4, and it finds exactly the same set of features as does an exhaustive search, but it requires far less memory.

One may also ask how this learning algorithm compares with other unsupervised "feature" detection algorithms. First, as has been discussed, this approach is related to Hinton and Pearlmutter's: both favor features with large P_f and with $P_f >> \prod P_w$, although theirs is not guaranteed to find a factorial code. The greater distinction is between algorithms that use these criteria and algorithms of the type proposed by von der Malsberg (1973) and by Bienenstock *et al.* (1982). Those also favor features with large P_f, *but* they prefer features composed of elements with *large* P_w. This may lead to features with small mutual information, and thus may include false background elements. For words in text this leads to poor segmentation, since many very tightly bound words are composed of relatively rare subwords.

6 Noise and Generalization

As mentioned in the introduction, desirable input information is often encoded redundantly (e.g., words in text) so redundancy can be used to distinguish true signal from noise. This is the case for example when the noise is not correlated with the true signal or with itself. Then the feature detection algorithm still finds the true signal redundancy—the true signal statistics—even though the total signal is noisy.

To show this, consider an English text with random noise, that is, a certain fraction of letters, chosen randomly, are incorrect. Taking the same sample text used in Section 4, but with 1/13 letters randomly incorrect, I applied the same algorithm as before. The result, shown in Figure 3, is that only real words and word combinations are chosen by the algorithm, while noisy letters are ignored. So noise does not confuse the feature detection. Once the features have been found, the text can be restored by using the probabilities of the redundant words to predict the noise-incorrect letters, that is, to build Bayesian filters.

It should be noted that in order to reconstruct the true text, one needs to know more than just the statistical properties of the noisy input messages. In the above example, one additionally needs to know that the

a l i c e w a s b e g i u n i n g t o g e t v e r y t i r e d o f s i t t i n g b y h e r s i s t e r o n
t h e b a n k a n d o f h a r i n g n o t h i t g t o d o o n c e o r t w z k e s h e h a d p e e p
e d i k t o z h e b o o k h e r s i s t e r w a s r e a d i n g b u t i t h a d n o p i l t u r e s b r
c b n v e r s a t i o n s i n i t a n d w h v t i s t h e u s e o f a b o o k t h o u g h t a l i c e w
i t h o u t p i q t u r e s o s c o n v e r s a t i o n s s r s h e w a s c o n s i d e r i n g f n h e
r o w n m i o d a s w e c l a s s h e c o u l d f o r t h e h o t d a y m a d e h e r f e e l v e r y
s l e e p y a n d s t u p i d w h e t h f r t h e p l e a s u r e o f m a i i n g a d a h s u c h a i
g w o u l d b e w o r t h t h e t r o u b l j o f g e t t i n g u p a n z p i c k i n g t h e d a q s i
e s w h e n s u d d e n l y a w h i n e r a b b i t w i t h p i n k e y e s r a n c l o s e b a h e
r t h e r e w a s n o t h i n g s o d e r y r e m a r k a b l e i n t h a t n o r d i d a l i c e t h i
n e i t s o v e r j m u a h o u t o f t h e w a y t o h e a r t h e r c b b i t s a y t o i t s e l f o
h d e a r o h d e a s

(a)

alice was beg i u ning toget verytiredof sitting by hersister onthe bank and of ha r
ing no th it g todo onceortw z k e shehad peeped i k to z he book hersister was read
ing but ithad no pil tures br c b n ver sation s in it and wh v t i s theuseof a book
thoughtalice without p i q tures o s conversation s s r shewas considering f n her
own m i o d a swec las shecould for the hot day madeher feelvery sleepyand stupi d
whe th f r the plea sure of m a i ing a d a h sucha i g wouldbe worth thetr ou b l j of
getting up an z pick ing the d a q s i e s whensudden ly a wh in e rabbit with p in key
e s r anc los e b a her therewas nothing s o der y remark able in that nor d idalice
thine its over j m u a h outoftheway tohear ther c bbit say toitself ohdear oh d e a s

(b)

Figure 3: Again, the same sample of text as in Figure 1, but with one out of 13 letters randomly incorrect. The noisy text before any redundancy reduction is shown in (a); it has $H_L = 4.26$ bits, which is slightly higher than the original text because it is less correlated. One of the later stages in redundancy reduction is shown in (b); it has entropy per letter $H_W/\bar{s} = 2.99$ bits. Note that the noise does not confuse the algorithm into finding false words or word combinations.

noise is *random*. In other words, one needs at least some outside knowledge or *supervision*. For example, mean squared filtering that uses the autocorrelator of an ensemble to filter out noise, can be implemented through a *supervised* perceptron-type algorithm (see Atick and Redlich 1990b).

This leads to an important point: purely *unsupervised* learning based strictly on statistics, does not lead to *conceptualization*. This is due to the implicit assumption that every distinguishable input state potentially carries a different message. In conceptualizing, on the other hand, different input states which carry the same *useful* message are grouped together. This grouping requires some further knowledge that distinguishes signal from noise, or provides a measure of closeness on the signal space (Kohonen 1984), or provides active supervision as in perceptron learning. Also, the information that distinguishes between different members of a concept can be thrown away, as in noise filtering. Since this *information reduction* effectively lowers the number of input states, it also simplifies the problem of learning and storing statistics. So one challenge is to in-

corporate in the present redundancy reduction strategy a controlled or supervised information reduction. Some first steps in this direction have been taken by Linsker (1989) and by Atick and Redlich (1990a), both using the mutual information between the desired scene data and the noisy image signal (for a different application of redundancy reduction to supervised learning, see Redlich 1992).

Acknowledgments

I thank J. Atick for his very perceptive comments on the manuscript. Also, this work was supported in part by a grant from the Seaver Institution and in part by the DOE DE-FG02-90ER40542.

References

Atick, J. J., and Redlich, A. N. 1990a. Towards a theory of early visual processing. *Neural Comp.* **2**, 308–320.

Atick, J. J., and Redlich, A. N. 1990b. Predicting ganglion and simple cell receptive field organizations. *Int. J. Neural Syst.* **1**, 305.

Atick, J. J., and Redlich, A. N. 1991. Convergent algorithm for sensory receptive field development. *Neural Comp.* In press.

Atick, J. J., and Redlich, A. N. 1992. What does the retina know about natural scenes? *Neural Comp.* **4**, 196–210.

Attneave, F. 1954. Some informational aspects of visual perception. *Psychol. Rev.* **61**, 183–193.

Barlow, H. B. 1961. Possible principles underlying the transformation of sensory messages. In *Sensory Communication*, W. A. Rosenblith, ed. MIT Press, Cambridge, MA.

Barlow, H. B. 1989. Unsupervised learning. *Neural Comp.* **1**, 295–311.

Barlow, H. B., and Foldiak, P. 1989. In *The Computing Neuron*. Addison-Wesley, New York.

Bienenstock, E. L., Cooper, L. N., and Munro, P. W. 1982. Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex. *J. Neurosci.* **2**, 32–48.

Carroll, L. 1865. *Alice in Wonderland*. Castle, Secaucus.

Eriksson, K., Lindgren, K., and Mansson, B. A. 1987. *Structure, Context, Complexity, Organization*, Chap. 4. World Scientific, Singapore.

Geman, S., and Geman, D. 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Trans. Patt. Anal. Machine Intell.* **PAMI-6**, 721–41.

Hinton, G. E., and Pearlmutter, B. A. 1986. G-maximization: an unsupervised learning procedure for discovering regularities. In *Neural Networks for Computing, AIP Conference Proceedings, Snowbird, UT*, J. S. Denker, ed. AIP Press, New York.

Hinton, G. E., and Sejnowski, T. J. 1983. Optimal perceptual inference. *Proceedings of the IEEE Conference on Computer Vision Pattern Recognition*, 448–453.

Kersten, D., 1990. Statistical limits to image understanding. In *Vision: Coding and Efficiency*, C. Blakemore, ed. Cambridge University Press, Cambridge.

Kohonen, T., 1984. *Self Organization and Associative Memory*, Springer-Verlag, Berlin.

Kullback, S., 1959. *Information Theory and Statistics*. Wiley, New York.

Linsker, R. 1989. An application of the principle of maximum information preservation to linear systems. In *Advances in Neural Information Processing Systems*, D. S. Touretzky, ed., Vol. 1, pp. 186–194. Morgan Kaufmann, San Mateo, CA.

Redlich, A. N. 1992. Supervised factorial learning. Preprint.

Shannon, C. E., and Weaver, W. 1949. *The Mathematical Theory of Communication*. The University of Illinois Press, Urbana.

Singh, J., 1966. *Great Ideas in Information Theory, Language and Cybernetics*, Chap. 16, Dover, New York.

Uttley, A. M., 1979. *Information Transmission in the Nervous System*. Academic Press, London.

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

Received 12 December 1991; accepted 29 September 1992.

This article has been cited by:

1. Jürgen Schmidhuber. 2014. Deep learning in neural networks: An overview. *Neural Networks* . [CrossRef]
2. Harald Hammarström, Lars Borin. 2011. Unsupervised Learning of Morphology. *Computational Linguistics* **37**:2, 309-350. [Abstract] [PDF] [PDF Plus]
3. Yuanlong Yu, George K I Mann, Raymond G Gosine. 2010. An Object-Based Visual Attention Model for Robotic Applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **40**, 1398-1412. [CrossRef]
4. Jim W. Kay, W. A. Phillips. 2010. Coherent Infomax as a Computational Goal for Neural Systems. *Bulletin of Mathematical Biology* . [CrossRef]
5. S. M. Boker, J. F. Cohn, B.-J. Theobald, I. Matthews, T. R. Brick, J. R. Spies. 2009. Effects of damping head movement and facial expression in dyadic conversation using real-time facial expression tracking and synthesized avatars. *Philosophical Transactions of the Royal Society B: Biological Sciences* **364**:1535, 3485. [CrossRef]
6. M.A. Sanchez-Montanes, F.J. Corbacho. 2004. A New Information Processing Measure for Adaptive Complex Systems. *IEEE Transactions on Neural Networks* **15**, 917-927. [CrossRef]
7. Mark J. Schnitzer, Markus Meister. 2003. Multineuronal Firing Patterns in the Signal from Eye to Brain. *Neuron* **37**, 499-511. [CrossRef]
8. F Pulvermüller. 2002. A brain perspective on language mechanisms: from discrete neuronal ensembles to serial order. *Progress in Neurobiology* **67**, 85-111. [CrossRef]
9. E Batchelder. 2002. Bootstrapping the lexicon: A computational model of infant speech segmentation. *Cognition* **83**, 167-206. [CrossRef]
10. J. Michael Herrmann. 2001. Dynamical systems for predictive control of autonomous robots. *Theory in Biosciences* **120**, 241-252. [CrossRef]
11. F Pulvermüller. 2001. Memory Traces for Words as Revealed by the Mismatch Negativity. *NeuroImage* **14**, 607-616. [CrossRef]
12. M Brent. 1999. Speech segmentation and word discovery: a computational perspective. *Trends in Cognitive Sciences* **3**, 294-301. [CrossRef]
13. Sepp Hochreiter, Jürgen Schmidhuber. 1999. Feature Extraction Through LOCOCODE. *Neural Computation* **11**:3, 679-714. [Abstract] [PDF] [PDF Plus]
14. Nick Chater. 1999. The Search for Simplicity: A Fundamental Cognitive Principle?. *The Quarterly Journal of Experimental Psychology Section A* **52**, 273-302. [CrossRef]
15. D. Obradovic, G. Deco. 1998. Information Maximization and Independent Component Analysis: Is There a Difference?. *Neural Computation* **10**:8, 2085-2101. [Abstract] [PDF] [PDF Plus]
16. Martin Redington, Nick Chater. 1998. Connectionist and Statistical Approaches to Language Acquisition: A Distributional Perspective. *Language and Cognitive Processes* **13**, 129-191. [CrossRef]

17. J Kay. 1998. Contextually guided unsupervised learning using local multivariate binary processors. *Neural Networks* **11**, 117-140. [CrossRef]

18. Jean-Pierre Nadal, Nestor Parga. 1997. Redundancy Reduction and Independent Component Analysis: Conditions on Cumulants and Adaptive Approaches. *Neural Computation* **9**:7, 1421-1456. [Abstract] [PDF] [PDF Plus]

19. G. Deco, L. Parra. 1997. Non-linear Feature Extraction by Redundancy Reduction in an Unsupervised Stochastic Neural Network. *Neural Networks* **10**, 683-691. [CrossRef]

20. Virginia R. de Sa, Dana H. BallardPerceptual Learning from Cross-Modal Feedback 309-351. [CrossRef]

21. J. Gerard Wolff. 1995. Computing as compression: An overview of the SP theory and system. *New Generation Computing* **13**, 187-214. [CrossRef]

22. Gustavo Deco, Bernd Schürmann. 1995. Learning time series evolution by unsupervised extraction of correlations. *Physical Review E* **51**, 1780-1790. [CrossRef]

23. G. Deco, W. Finnoff, H. G. Zimmermann. 1995. Unsupervised Mutual Information Criterion for Elimination of Overtraining in Supervised Multilayer Networks. *Neural Computation* **7**:1, 86-107. [Abstract] [PDF] [PDF Plus]

24. G. Deco, D. Obradovic. 1995. Linear redundancy reduction learning. *Neural Networks* **8**, 751-755. [CrossRef]

25. G Deco. 1995. Nonlinear higher-order statistical decorrelation by volume-conserving neural architectures. *Neural Networks* **8**, 525-535. [CrossRef]

26. A. Norman Redlich. 1993. Supervised Factorial Learning. *Neural Computation* **5**:5, 750-766. [Abstract] [PDF] [PDF Plus]