

Randomly connected sigma-pi neurons can form associator networks

Tony A. Plate

Bios Group Inc,
317 Paseo de Peralta,
Santa Fe, NM, 87501, USA

E-mail: tony.plate@biosgroup.com

Abstract.

A set of sigma-pi units randomly connected to two input vectors forms a type of hetero-associator related to convolution- and matrix-based associative memories. Associations are represented as patterns of activity rather than connection strengths. Decoding the associations requires another network of sigma-pi units, with connectivity dependent on the encoding network. Learning the connectivity of the decoding network involves setting n^3 parameters (where n is the size of the vectors), and can be accomplished in approximately $3e n \log n$ presentations of random patterns. This type of network encodes information in activation values rather than in weight values, which makes the information about relationships accessible to further processing. This accessibility is essential for higher-level cognitive tasks such as analogy processing. The fact that random networks can perform useful operations makes it more plausible that these types of associative networks could have arisen in the nervous systems of natural organisms during the course of evolution.

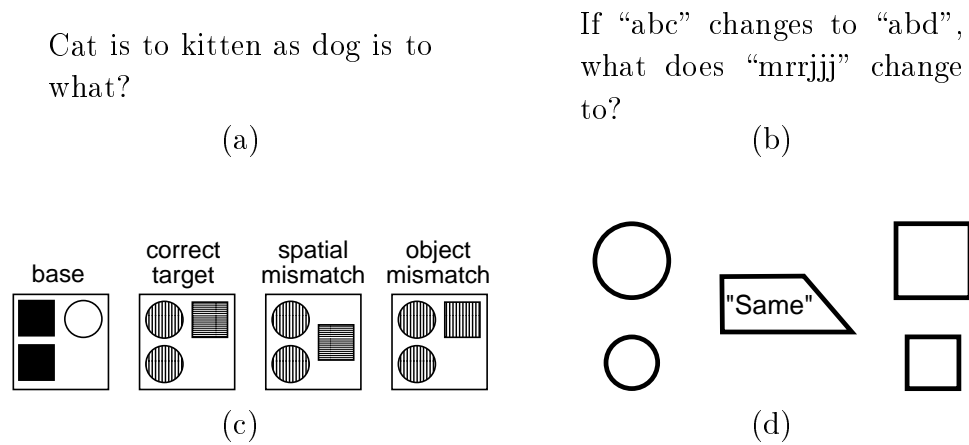


Figure 1. Analogy tasks. (a) is an example of the common verbal analogies of the type A is to B as C is to what? (b) is a task solved by the COPYCAT system (Hofstadter and Mitchell 1994). (c) is a task presented to human subjects in a PET scan study intended to identify regions of the brain involved in processing analogies. The task was to choose the arrangement analogous to the base arrangement. (Wharton, Grafman, Flitman, Hansen, Bruaner, Marks, and Honda 1998). (d) is a task presented to chimpanzees; the task was to place the symbol for “same” or “different” in the center of the figure, depending on whether the relationship between the pair of objects on the left is the same as the relationship between the pair of objects on the right (Oden, Thompson, and Premack 1998).

1. Introduction

The ability to reason about knowledge itself is one of the hallmarks of higher-level cognitive processing. The knowledge structures involved can range from simple associations or relations, e.g., as in the analogy tasks shown in Figure 1, to the rules of reasoning itself. Any theory of higher-level cognitive processing must eventually address how complex knowledge structures can be represented, combined, compared, and otherwise processed.

One candidate theory of higher-level cognitive processing begins with the ideas that concepts can be represented as distributed patterns of neuronal activity, and that relationships between concepts can be represented as associations in an associative memory (Willshaw, Buneman, and Longuet-Higgins 1969; Hinton 1989; Rumelhart, McClelland, and the PDP research group 1986). However, when we consider the requirement of higher-level processing that relationships themselves must be examinable, a problem immediately arises: knowledge about relationships is hidden in the weights of the network and is inaccessible to further analysis or processing. This initial problem is easily overcome by formulating associative memories so that associations are encoded in activation values rather than weights. These types of associative memories are more like an adder or arithmetic logic unit in a von Neumann computer than like traditional memory. They can be used dynamically to create new knowledge structures from old ones rather than merely storing knowledge as static associations for later retrieval.

Because associations are represented in patterns of activity, they are available for processing, allowing higher-level tasks, such as analogies, to be accomplished (Halford, Wilson, and Phillips 1998; Gayler 1998; Kanerva 2000; Plate 1994a; Plate 2000).

However, these memories that store associations in activation patterns face a more serious plausibility gap: it is difficult to see how the intricate and precise patterns of connectivity they utilize could have evolved in natural organisms. This paper shows that associative memories actually do not require intricate and precise patterns of connectivity. In fact, a randomly connected network of sigma-pi units, with some conditions on the density and nature of connections, turns out to constitute the encoding half of a hetero-associative memory – it can encode the association of two patterns. The corresponding network required to decode the associations (i.e., to retrieve one pattern given the other) has complementary connections, which can be learnt by a simple algorithm in a reasonable amount of time.

2. Associative memories

Pairwise associations between patterns are an interesting form of knowledge because they are both richly productive and simple. Various authors, e.g., Plate (1995), Plate (2000), Gayler (1998), and Kanerva (2000), have shown how pairwise associations, encoded using any one of a variety of associative memory schemes, can be used as the basis for representing and processing more complex knowledge, allowing analogy problems like those in Figure 1 to be solved.

Pairwise associations can be learned by many types of neural networks, including feedforward networks trained via backpropagation (slow learning) (Rumelhart, Hinton, and Williams 1986) and hetero-associative memory networks (one-shot learning), such as Willshaw, Buneman, and Longuet-Higgins’s (1969) associative network. In both feedforward and associative networks, the knowledge about associations can be *used* to produce one element of a pair given the other, but cannot be *examined or manipulated*. In order for knowledge to be manipulated, e.g., passed to other networks for further processing, it must be encoded in the activation values of neurons rather than in the connection strengths. Such a pattern of activations that can encode a number of associations is referred to in this paper as a memory *trace*.

It is possible to formulate auto- and hetero-associative memory networks so that associations are encoded in activation values rather than weights. This requires neurons to be connected in a sparse and very precise and regular fashion. For example, matrix-based memories, such as Willshaw *et al*’s (1969) associative nets and Smolensky’s (1990) tensor product memories, can be formulated as a network of sigma-pi neurons that are connected in patterns corresponding to the computation of an outer-product. (Sigma-pi units compute a sum of products of inputs.) Convolution-based memories such as Willshaw, Buneman, and Longuet-Higgins’s (1969) non-linear correlograph, Murdock’s (1982) TODAM, and Plate’s (1995) Holographic Reduced Representations (HRRs) are naturally formulated in terms of sigma-pi neurons where information about

associations is encoded in activations rather than weights. The required connectivity of neurons corresponds to correlation or convolution formulas. Thus, both matrix- and convolution-based schemes require precise patterns of interconnection, and in the convolution-based schemes the patterns are intricate as well as precise.

Willshaw *et al*'s (1969) non-linear correlograph is one of the first examples of associative memories in the literature (see also Willshaw (1989)). It encodes associations between binary patterns into binary traces, and uses correlation and convolution formulas for encoding and decoding associations. Like other convolution/correlation-based memories, it involves intricate and precise arrangements of connections. As the association networks described in this paper can be seen as a randomly-connected version of the non-linear correlograph, it is useful to review the characteristics of the non-linear correlograph.

Willshaw, Buneman, and Longuet-Higgins (1969) show that the maximum information capacity of the non-linear correlograph is $0.693n$, where n is the number of bits in both patterns and traces. This means that the non-linear correlograph can function at an information efficiency of 69%, since n bits of information are stored in the trace. This maximum information capacity is achieved when $0.693n/(\log_2 n)^2$ pairs of very sparse patterns are stored; patterns have a density of $m = \log_2 n$ bits, i.e., only $\log_2 n$ out of n bits are on.

This is the same information efficiency (measured in information stored per parameter) as Willshaw *et al*'s better-known associative network, a matrix-based associative memory with binary weights. The associative network has a higher capacity for a given vector size, reflecting its corresponding higher resource requirements. It can store more pairs of patterns of a given size because it has n^2 binary weights rather than the n binary parameters in the non-linear correlograph.

3. Random Sigma-Pi Associators

3.1. Encoding network

In the non-linear correlograph the elements of the trace and the elements of the decoded pattern can be computed by sigma-pi units: they are both thresholded sums of products. The connectivity of these sigma-pi units is highly ordered: the k th unit of the trace is computed as

$$z_k = h \left(\sum_{j=0}^{n-1} x_j y_{k-j} \right) \quad (1)$$

where $h(x)$ is the threshold function $h(x) = 1 \ \forall \ x > 0$ and $h(x) = 0$ otherwise.

Consider what happens when the correlation formula for encoding in the non-linear correlograph is replaced by a sum of random products:

$$z_k = h \left(\sum_{ij} w_{kij} x_i y_j \right) \quad (2)$$

the perfectly ordered binary correlograph. The threshold θ is a global parameter, but could just as well be local to each output neuron of the decoding network. In either case, it could be learned using the perceptron rule or delta rule for single layer networks (Hertz, Krogh, and Palmer 1991).

3.3. Decoding Accuracy

The probability of correctly decoding associations in the trace of a random sigma-pi associator can be worked out with similar steps as for the non-linear correlograph. Suppose as before that \mathbf{t} is the trace resulting from encoding R pairs of patterns, including the pair \mathbf{x} and \mathbf{y} . Let \mathbf{s} be the pattern resulting from decoding \mathbf{z} with \mathbf{y} , i.e., $\mathbf{s} = \text{decode}(\mathbf{z}, \mathbf{y})$. We would like $\mathbf{s} = \mathbf{x}$.

First, consider the density of the pattern \mathbf{z} resulting from associating \mathbf{x} and \mathbf{y} , i.e., $\mathbf{z} = \text{encode}(\mathbf{x}, \mathbf{y})$. Note that z_i is the sum of products like $x_j y_k$. There are n^2 such products, of which m^2 are 1, and each has a G/T chance of being in the sum for z_i . Thus, the probability that z_i is 0 is $(1 - (Gm^2)/(Tn^2))^{n^2}$. Using the same approximation as before (i.e., $(1 - \epsilon)^x \approx \exp(-x\epsilon)$ for small ϵ), we get $\Pr(z_i = 0) \approx \exp(-Gm^2/T)$ and

$$\Pr(z_i = 1) \approx 1 - \exp\left(\frac{-Gm^2}{T}\right). \quad (4)$$

Now consider \mathbf{t} , which is the logical-OR of R patterns like \mathbf{z} . The probability that t_i is zero is

$$\Pr(t_i = 0) \approx \exp\left(\frac{-RGm^2}{T}\right). \quad (5)$$

Then, p , the probability that a particular t_i is 1 is

$$p \approx 1 - \exp\left(\frac{-RGm^2}{T}\right). \quad (6)$$

Note that for $T = n$ (a trace with n elements) and $G = 1$ (an average of n products in the sum for each trace), this is the same trace density as for the binary correlograph.

Next, consider a particular s_i , which should have the same value as x_i . Recall that $s'_i = \sum_{jk} v_{ijk} y_j t_k$. There are Tn terms in this sum, and the individual factors in these terms are binary events with the following probabilities: $\Pr(v_{ijk} = 1) = G/T$, $\Pr(y_j = 1) = m/n$, and $\Pr(t_k = 1) = p$. Consider the cases of $x_i = 0$ and $x_i = 1$ separately.

Case 1: $x_i = 0$. Let μ_0 be the mean of s'_i when $x_i = 0$. When $x_i = 0$ the factors in the terms for s'_i are uncorrelated and s'_i is the sum of Tn random binary values, each with probability $G/T \cdot m/n \cdot p$. Thus, s'_i has a binomial distribution with $\mu_0 = pmG$.

Case 2: $x_i = 1$. Let μ_1 be the mean of s'_i when $x_i = 1$. Consider a term $v_{ijk} y_j t_k$ in s'_i for which $v_{ijk} = 1$ and $y_j = 1$ (terms for which either of these two conditions are not true do not contribute to s'_i). This means that $w_{kij} = 1$, and thus t_k is also 1. Thus, s'_i is the sum of Tn random binary values, each with probability $G/T \cdot m/n$, and has a binomial distribution with $\mu_1 = mG$.

Parameters of net			Trace density		Elements with target 1		Elements with target 0		Separation (in σ 's)		Threshold	Number of errors							
n	m	R	p		μ_1	σ_1	μ_0	σ_0	d		θ	E_1 (of m)		E_0 (of $n-m$)					
1024	10	2	0.177	0.178	10	9.98	3.16	3.15	1.77	1.78	1.33	1.34	1.83	1.83	5	0.66	0.66	9.78	10.1
4096	10	3	0.071	0.071	10	10.03	3.16	3.09	0.71	0.71	0.84	0.84	2.32	2.37	5	0.67	0.61	0.38	0.38
4096	15	3	0.152	0.152	15	15.01	3.87	3.74	2.28	2.28	1.51	1.43	2.36	2.46	8	0.58	0.45	2.44	2.33
4096	23	3	0.321	0.321	23	22.94	4.79	4.65	7.39	7.38	2.72	2.66	2.08	2.13	14	0.71	0.66	36.6	34.4

Table 1. Analytically derived and experimentally measured (in italics) means and standard deviations of the pre-threshold elements of the reconstructed pattern (i.e., s'_i) for elements with a target of zero and elements with a target of one. The number of errors is per reconstructed vector, for a threshold chosen to make the expected number of errors in one-bits (places of the reconstructed vector with a target of one) just less than one.

The reconstruction of \mathbf{x} will be accurate if the average value of s'_i is low where $x_i = 0$, and high where $x_i = 1$, and the threshold is located somewhere between their means. Let the mean of s'_i where $x_i = 0$ be μ_0 and the mean of s'_i where $x_i = 1$ be μ_1 . Both of these distributions are binomial and can be approximated by a normal distribution. Under the normal approximation the variances are $\mu_0(1-\mu_0/n)$ and $\mu_1(1-\mu_1/n)$. As μ_0 and μ_1 are small relative to n , we can drop the $(1-\mu/n)$ factors. Let d be the distance, measured in average standard deviations, between the means of the distributions of s'_i 's corresponding to zero and non-zero elements (i.e., μ_0 and μ_1). Then, using the normal approximations, we can calculate d as follows:

$$\begin{aligned}
d &\approx \frac{\mu_1 - \mu_0}{\frac{1}{2}(\sigma_1 + \sigma_0)} \approx \frac{2(1-p)}{1 + \sqrt{p}} \sqrt{mG} \approx \frac{2 \exp(-\frac{RGm^2}{T})}{1 + \sqrt{1 - \exp(-\frac{RGm^2}{T})}} \sqrt{mG} \\
&= 2c\sqrt{mG}, \quad \text{where } c = \frac{(1-p)}{1 + \sqrt{p}} = \frac{\exp(-\frac{RGm^2}{T})}{1 + \sqrt{1 - \exp(-\frac{RGm^2}{T})}} \quad (7)
\end{aligned}$$

In order to test the accuracy of these approximations, some values of parameters, and some values of p , μ 's and σ 's were measured from simulations. These are compared with the analytically calculated values in Table 1. The experimentally measured reconstruction accuracy agrees closely with the analytically derived accuracy in all respects. In all of these cases, $T = n$ (i.e., the trace was the same size as the patterns being associated) and $G = 1$ (i.e., there was an average of n products involved in the sum for each element of the trace). The experimentally measured values, shown in italics, were computed over 1000 storage and decoding operations in a single randomly constructed network. E_1 and E_0 are the average number of errors in a reconstructed pattern for 1's and 0's respectively. The analytically calculated values for E_1 and E_0 were calculated using the binomial cumulative probability function. The thresholds used for both the analytic and experimental figures were the highest thresholds that resulted in less than one error in reconstructing 1's (analytically).

Some analysis of the formula for the separation of μ_0 and μ_1 can help identify the usable region of the parameter space of n, m, G, T, R . Consider c in the formula for d

in Equation 7. The lefthand plot in Figure 3 shows that c varies between 0 and 1 and is greater than 0.5 for approximately $RGm^2/T < 0.25$. The exponential dropoff of c suggests that in order to achieve anywhere near reasonable separation with $G = 1$, we should have $m \geq 10$ (so that $\sqrt{mG} > 3$), and $T > RGm^2$ (so that $c > 0.5$), thus giving $d > 3$. For fixed R , G , and T , separation is at a maximum for moderately small values of m , as shown in the right-hand plot in Figure 3.

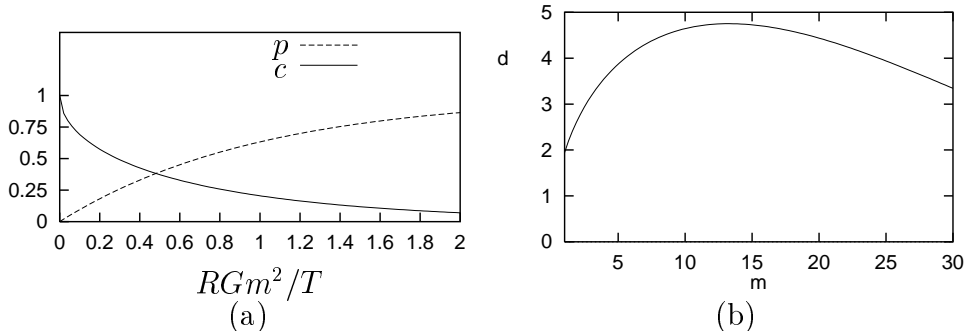


Figure 3. (a) Behavior of p and c in the equation for the separation (Eqn 6 and Eqn 7). (b) The separation d (Eqn 7) plotted against m for $R = 3$, $G = 1$, and $T = 4096$.

4. Learning the Decoding Network

How could the functional structure of the decoding network be constructed in a neurally-plausible manner? As explained in Section 3.2, if the connectivity of the encoding network is defined by an assignment of 0's and 1's to w_{kij} (a “1” means that the product $x_i y_j$ of elements of the input patterns is involved in the sum for the k 'th element of the trace pattern; and a “0” means that the product $x_i y_j$ is not involved in that sum for the k 'th element of the trace pattern), then the corresponding decoding network has connectivity similarly defined by v_{ijk} , with $v_{ijk} = w_{kij}$. Although this decoding connectivity maps onto the encoding connectivity if connections are reversed, connections in biological neural networks are one-way, so propagating activation values backwards through the encoding network is not plausible. Thus, a separate network must be used. This decoding network must be accurately set up to correspond to the encoding network. It turns out that a suitable decoding network can be learnt using a simple rule in a number of training passes that is surprisingly small given the number of parameters to be learnt.

The learning begins with a completely connected decoding network, i.e., $v_{ijk} = 1 \forall ijk$, and proceeds by setting the v 's to 0 based on the presentations of random patterns. First, generate random patterns \mathbf{x} and \mathbf{y} . Second, put these through the encoding network to obtain pattern $\mathbf{z} = \text{encode}(\mathbf{x}, \mathbf{y})$. Finally, present the pair \mathbf{y}, \mathbf{z} to the inputs of the decoding network, and \mathbf{x} to the outputs of the decoding network. If x_i and y_j are 1, and z_k is 0, then set v_{ijk} to 0 (because w_{kij} must be 0, otherwise z_k would be 1). Since $\Pr(z_k = 0) = \exp(-Gm^2/T)$ and $\Pr(x_i = 1) = \Pr(y_j = 1) = m/n$,

the probability that we can set v_{ijk} to zero (if it should be) is $m^2/n^2 \exp(-Gm^2/T)$. This rule will never set v_{ijk} to 0 when it should be 1. Thus the probability that v_{ijk} has the wrong value at the end of Q trials (i.e., Q presentations of random patterns) is as follows:

$$\left(1 - \frac{m^2}{n^2} \exp\left(-\frac{Gm^2}{T}\right)\right)^Q.$$

We would like this to be small, say $\frac{1}{Tn^2}$, so that on average only one of the v_{ijk} is wrong after Q learning trials. Using the approximation $\log(1 - \epsilon) = -\epsilon$ for small ϵ , we get

$$Q = \frac{n^2 \log(n^2 T) \exp(Gm^2/T)}{m^2}. \quad (8)$$

For $n = 4096$, $m = 15$, $G = 1$, and $T = 4096$, this gives a learning time of 1.97 million trials.

However, the learning time depends on m , and a higher m (i.e., greater density of ones in \mathbf{x} and \mathbf{y}) can result in faster training. To find the value of m which minimizes the learning time Q , we can first find the derivative of Q with respect to m :

$$\frac{dQ}{dm} = \frac{2n^2}{m} \log(n^2 T) \exp\left(\frac{Gm^2}{T}\right) \left(\frac{G}{T} - \frac{1}{m^2}\right). \quad (9)$$

Thus Q has a minimum at $m = \sqrt{T/G}$, at which point we have

$$Q = n^2/T \log(n^2 T)e \quad (10)$$

When $G = 1$ and $T = n$, the minimum number of learning trials is $3 G n e \log n$, which for the example of $n = 4096$ is 280,000 learning trials. Considering there are $4096^3 = 6.8 \times 10^{10}$ parameters to be set, this is a small number of trials for a one-off exercise. A system that learned from encoding and decoding internally generated random patterns (with the optimal density for learning) at the rate of one pair of patterns per second could learn the decoding connectivity for this size network in 78 hours.

5. Comparison to Matrix- and Convolution-Based Associators

Both matrix and convolution-based associators are special cases of sigma-pi associators, with highly regular connectivities. Convolution-based associators have $T = n$, $G = 1$, and $w_{kij} = 1$ if $j = k - i \bmod n$. Each product $x_i y_j$ occurs once in only one z_k , each z_k is the sum of exactly n products, and x_i and y_j appear in exactly one product in each z_k . Matrix-based associators have $T = n^2$, $G = 1$, and $w_{kij} = 1$ if $k = in + j$. Each of the $n^2 z_k$'s consists of a single product, and there is one for each $x_i y_j$ pair. The parameters and properties of these various styles of associators are summarized in Table 2.

The highly ordered nature of matrix- and convolution-based memories results in more accurate reconstructions and higher information capacity than their random cousins. For example, a non-linear correlograph with $n = 4096$ and $m = 12$ can encode associations between 19 pairs of patterns and have an average of less than 1 error

	Associative net (Matrix-style)	Convolution/ Correlation	Random Sigma-Pi
In general			
T (size of trace)	n^2	n	n (can vary)
$G = (\text{total \# of products in trace})/n^2$	1	1	1 (can vary)
\# of products involved in each element of trace	1	n	n on average (varies with G & T)
$w_{kij} = 1 \dots$ encoding connectivity	iff $k = in + j$	iff $j = k - i \text{ mod } n$	with prob. $1/n$ (prob. varies with G & T)
m (\# of 1's in patterns) (optimized for info. efficiency)	$\log_2 n$	$\log_2 n$	$> \log_2 n$
Max. information efficiency	69%	69%	$\ll 69\%$
Information efficiency in a particular example			
n (size of patterns)	4,096	4,096	4,096
m (\# of 1's in patterns)	12	12	15
T (size of trace)	16,777,216	4,096	4,096
p (density of 1's in trace)	0.5	0.5	0.152
R (\# of pairs)	80,757	19	3
Information stored (bits) (approximate)	11,629,080	2,736	540
Information efficiency	69%	69%	13.2%

Table 2. Comparison of Willshaw *et al*'s (1969) associative net (a matrix-style associative memory with binary traces), a convolution/correlation associator (a non-linear correlograph), and random sigma-pi associators. The equations underlying the numbers given for the associative net are from Willshaw *et al*.

in reconstructed 0's (all the 1's are guaranteed to be reconstructed correctly). This corresponds to the storage of about 2736 bits in a trace of 4096 bits. In contrast, a random sigma-pi associator with $n = 4096$, $G = 1$, and $T = 4096$ (i.e., the same pattern size, trace size, and connection density, but with random connectivity) can encode associations between 3 pairs of patterns with $m = 15$ and have an average of less than 1 error in reconstructed 0's and 1's. This corresponds to the storage of about 540 bits of information in a trace of 4096 bits. Thus, for this size of associator, the ordered nature of the connections in the non-linear correlograph appears to give about a 5-fold increase in information efficiency.

6. Storage of Dynamic and Structured Information

Random sigma-pi networks constitute a workable (though not not optimally efficient) network that can encode associations in the activation values of neurons. They can be

used to encode small chunks of knowledge, e.g., several pairwise associations, in a form that is amenable to being stored in and manipulated by other networks. These type of encoding networks are useful in the encoding of recursive structure and in encoding knowledge in such a way that it can be examined for further processing. That these random networks work at all is surprising and provides an avenue by which associative memory mechanisms that encode information in activations could have arisen in the brain. This avenue becomes even more plausible when one notices that the encoding part of the network (i.e., the random part) could be useful by itself, as a means for recording conjunctions of feature patterns, which in turn can support recognition processes (Mel and Fiser 2000).

In an extensive review of the information processing capabilities of dendritic trees, Mel (1994) comes to the conclusion that the computational capabilities of real neurons are more like sigma-pi units than thresholded linear units. Local effects within the dendritic tree can result in activation at nearby synapses having a non-linear effect. Overall, this can lead to the dendritic tree computing the sum of the boolean AND function of sets of nearby inputs. This suggests that the random sigma-pi associative memory networks discussed in this paper could map straightforwardly onto neural tissue. In considering how such networks could develop, several important questions arise. For the encoding networks, the major question is how the synapses could develop so that the sets of synapses involved in products always involved two connections: one from one pattern and one from the other. For the decoding network there are several further questions. How could a sufficiently rich set of possible connections be available? Is there a way that the learning rule for removing unnecessary connections in the decoding network (described in Section 4) could be implemented in real neural tissue?

Conventional convolution- and matrix-based memories are special cases of the random sigma-pi networks presented in this paper. Thus, sigma-pi networks provide a general framework for associative memories, with the connection density being one dimension of variation, and the orderedness of connectivity being another, independent, dimension of variation. In fact, the space of associative memory schemes that can be described as sigma-pi networks is surprisingly rich. If the sigma operation is extended to other types of multiplication, the space also includes several recently discovered schemes that require only element-wise multiplication of patterns: Plate's (1994a) frequency-domain HRRs, in which patterns have complex values on the unit circle, and Kanerva's (1996) binary spatter codes, which work with binary patterns. Both of these schemes have small G (the total number of products involved in the encoding, relative to the number required for convolution and outer-product encoders) and require only n operations, which can be performed in parallel, for encoding and decoding.

Storing associations in unit activations means that associations are, in the terminology of programming, first class objects. They can potentially be manipulated and processed in the same manner as other patterns, rather than being hidden in slowly changing connection strengths. This ability to dynamically process associations, and possibly involve them in further recursive associations, is essential in an information

processing system that must deal with dynamic and complex knowledge structures, as does the human brain. Various authors, including Murdock (1982), Smolensky (1990), Pollack (1990), Plate (1994a), Plate (1995), Plate (2000), Halford *et al* (1994), Halford *et al* (1998), Gayler (1998), and Kanerva (2000), have proposed schemes for encoding structured compositional knowledge in vector representations. These knowledge structures can represent diverse information such as complex relationships between entities, or the recursive phrase structure of language. All of these schemes, including the one presented in this paper, are able to support higher-level cognitive tasks such as analogy detection and processing tasks (see Plate (2000) for a discussion of some analogy tasks) because they share four fundamental properties: (1) associations can be encoded and decoded; (2) associations can be recursive; (3) association is similarity-preserving (if a is similar to a' in the sense that their cosine is high, and b is similar to b' , then the association of a and b will be similar to the association of a' and b'); and (4) association is difference-preserving (if a is not at all similar to c in the sense that their cosine is near zero, then the association of a and b will be not at all similar to the association of c and b').

There are other methods for storing information about complex structure (binding information) in networks, e.g., the schemes described by Shastri and Ajjanagadde (1993) and Hummel and Holyoak (1997) for representing binding through temporal synchrony. However, storing structures as static distributed representations seems particularly promising because it allows one to exploit the ease with which flat vector representations can be manipulated, and mappings between them learned, by neural networks, and to exploit the rich similarity relationships which are possible with distributed representations (Plate 1994b; Plate 2000; Kanerva 2000).

Once we know that rudimentary associative encoding networks can be built out of random networks of sigma-pi units, it seems more plausible that these or related schemes might be used for storing structured information in the human brain. The ease with which decoding networks can be learnt increases the plausibility. The possibility is made even more intriguing by Mel's (1994) claim that the predominant type of neuron in the brain computes something close to a sigma-pi function.

Acknowledgements

The author thanks Pentti Kanerva, Ross Gayler and the two anonymous reviewers for their detailed and very useful comments. Some of the work reported in this manuscript was carried out while the author was a faculty member at the School of Mathematics and Computing Sciences at Victoria University in Wellington, New Zealand.

References

- Gayler, R. W. (1998). Multiplicative binding, representation operators and analogy. In K. Holyoak, D. Gentner, and B. Kokinov (Eds.), *Advances in analogy research: Integration of theory and data from the cognitive, computational, and neural sciences*. Sofia, Bulgaria: New Bulgarian University. Only the abstract appears in the proceedings. Full text available at <http://cogprints.soton.ac.uk>.
- Halford, G., W. H. Wilson, and S. Phillips (1998). Processing capacity defined by relational complexity: Implications for comparative, developmental, and cognitive psychology. *Behavioral and Brain Sciences* 21(6), 803–831.
- Halford, G. S., W. H. Wilson, J. Guo, R. W. Gayler, J. Wiles, and J. E. M. Stewart (1994). Connectionist implications for processing capacity limitations in analogies. In K. J. Holyoak and J. Barnden (Eds.), *Analogical Connections*, Volume 2 of *Advances in Connectionist and Neural Computation Theory*. Norwood, NJ: Ablex.
- Hertz, J., A. Krogh, and R. G. Palmer (1991). *Introduction to the Theory of Neural Computation*. Santa Fe Institute studies in the sciences of complexity. Lecture notes ; v. 1. Redwood City, Calif.: Addison-Wesley Pub. Co.
- Hinton, G. E. (1989). Implementing semantic networks in parallel hardware. In G. E. Hinton and J. A. Anderson (Eds.), *Parallel Models of Associative Memory* (Updated ed.). Hillsdale, NJ: Erlbaum.
- Hofstadter, D. R. and M. Mitchell (1994). The copycat project: A model of mental fluidity and analogy-making. In K. J. Holyoak and J. A. Barnden (Eds.), *Analogical Connections. Advances in Connectionist and Neural Computation Theory, Vol. 2*, pp. 31–112. Norwood, NJ: Ablex.
- Hummel, J. E. and K. J. Holyoak (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review* 104(3), 427–466.
- Kanerva, P. (1996). Binary spatter-coding of ordered k-tuples. In C. von der Malsburg, W. von Seelen, J. Vorbruggen, and B. Sendhoff (Eds.), *Artificial Neural Networks—ICANN Proceedings*, Volume 1112 of *Lecture Notes in Computer Science*, Berlin, pp. 869–873. Springer.
- Kanerva, P. (2000). Large patterns make great symbols: An example of learning from example. In S. Wermter and R. Sun (Eds.), *Hybrid Neural Systems*, pp. 194–203. Heidelberg: Springer.
- Mel, B. W. (1994). Information processing in dendritic trees. *Neural Computation* 6(6), 1031–1085.
- Mel, B. W. and J. Fiser (2000). Minimizing binding errors using learned conjunctive features. *Neural Computation* 12, 247–278.
- Murdock, B. B. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review* 89(6), 316–338.

- Oden, D. L., R. K. R. Thompson, and D. Premack (1998). Analogical problem-solving by chimpanzees. In K. Holyoak, D. Gentner, and B. Kokinov (Eds.), *Advances in Analogy Research: Integration of Theory and Data from the Cognitive, Computational, and Neural Sciences.*, pp. 37–48. Sofia, Bulgaria: New Bulgarian University.
- Plate, T. (2000). Structured operations with vector representations. *Expert Systems: The International Journal of Knowledge Engineering and Neural Networks* 17(1), 29–40. Special Issue on Connectionist Symbol Processing.
- Plate, T. A. (1994a). *Distributed Representations and Nested Compositional Structure*. Ph. D. thesis, Department of Computer Science, University of Toronto. Available at <http://www.cs.utoronto.ca/~tap>.
- Plate, T. A. (1994b). Estimating structural similarity by vector dot products of holographic reduced representations. In J. D. Cowan, G. Tesauro, and J. Alspector (Eds.), *Advances in Neural Information Processing Systems 6 (NIPS*93)*, San Mateo, CA, pp. 1109–1116. Morgan Kaufmann.
- Plate, T. A. (1995). Holographic reduced representations. *IEEE Transactions on Neural Networks* 6(3), 623–641.
- Pollack, J. B. (1990). Recursive distributed representations. *Artificial Intelligence* 46(1-2), 77–105.
- Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). Learning internal representations by back-propagating errors. *Nature* 323, 533–536.
- Rumelhart, D. E., J. L. McClelland, and the PDP research group (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vols. 1 and 2*. Cambridge, MA: MIT Press.
- Shastri, L. and V. Ajjanagadde (1993). From simple associations to systematic reasoning: A connectionist representation of rules, variables and dynamic bindings using temporal synchrony. *Behavioral and Brain Sciences* 16(3), 417–494.
- Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence* 46(1-2), 159–216.
- Wharton, C. M., J. Grafman, S. K. Flitman, E. K. Hansen, J. Bruaner, A. Marks, and M. Honda (1998). The neuroanatomy of analogical reasoning. In K. Holyoak, D. Gentner, and B. Kokinov (Eds.), *Advances in Analogy Research: Integration of Theory and Data from the Cognitive, Computational, and Neural Sciences.*, pp. 260–268. Sofia, Bulgaria: New Bulgarian University.
- Willshaw, D. (1989). Holography, associative memory, and inductive generalization. In G. E. Hinton and J. A. Anderson (Eds.), *Parallel Models of Associative Memory* (Updated ed.), pp. 103–124. Hillsdale, NJ: Erlbaum.
- Willshaw, D. J., O. P. Buneman, and H. C. Longuet-Higgins (1969). Non-holographic associative memory. *Nature* 222, 960–962.