In M. Marinaro & P. G. Morasso (eds.), *ICANN '94, Proceedings of the International Conference on Artificial Neural Networks* (Sorrento, Italy). London: Springer–Verlag, 1994; 226–229.

The Spatter Code for Encoding Concepts at Many Levels

Pentti Kanerva
Swedish Institute of Computer Science
Box 1263, S–164 28 Kista, Sweden
e-mail: kanerva@sics.se

Abstract. The *Spatter Code* is a high-dimensional (e.g., N = 10,000), random code that encodes "high-level concepts" in terms of their "low-level attributes" so that concepts at different levels can be mixed freely. The binary spatter code is the simplest. It has two N-bit codewords for each concept or item, a "high-level," or *dense*, word with many randomly placed 1s and a "low-level," or *sparse*, word with a few (that are contained in the many). The dense codewords can be used as inputs to an associative memory. The sparse codewords are used in encoding new concepts. When several items (attributes, concepts, chunks) are combined to form a new item, the two codewords for the new item are made from the sparse codewords of its constituents as follows: the new dense word is the logical OR of the constituents (i.e., their sum thresholded at 0.5), and the new sparse word has 1s where the constituent words overlap (i.e., their sum thresholded at 1.5). When the parameters for the code are chosen properly, the number of 1s in the codewords is maintained as new items are encoded from combinations of old ones.

1 Introduction

In what follows, *mind*, *concept*, *attribute*, and *chunk* refer to philosophical or psychological entities, *brain* and *pattern* refer to physical entities, and *term* refers to something in between, but rather physical, for example, to a pattern of neural activity that has become established or organized through repetition and that can then be recreated by the brain.

The human mind is born without a rich repertoire of concepts, and the world is not organized into a neat system of objects, properties, and relations when the brain first "sees" it. The brain must find whatever organization there is in the world and it must construct the terms in which it sees the world. The human brain has a remarkable ability to do this; it is built to do it—it cannot help but to do it—which is why the organized world seem utterly natural to us by the time we are able to talk about it. But how the brain gets organized is a big mystery.

We do know some general principles. For example, the brain remembers recurring activity patterns and detects regularities in them. That is why things become familiar to us when we see them over and over, and why we rehearse when we want to learn a physical or a mental skill. Mathematical models for studying such things include associative memory and self-organizing maps (Hassoun, 1993; Kohonen, 1988;

Willshaw & von der Malsburg, 1975 & 1979).

The neural activity patterns most readily available in the new-born brain are produced by the sense organs and by emotions. They are the initial raw material, and whatever regularities are found in them, become the brain's initial terms for describing the world.

By age three the human mind has developed a rich store of concepts, built out of other concepts. This is expressed most strikingly in the use of language. The "high-level" terms that the brain now uses are built out of and on top of the brain's initial terms for describing the world, so that they form a kind of a superstructure.

This paper is about building the superstructure. Could it be built from the low-level terms in the same way as the initial, low-level terms are built from patterns of activity from the senses and emotions? For example, if a certain combination of low-level terms occurs repeatedly in some part of the brain, would a new higher level term be produced? It could be, if new activity patterns are generated and regenerated from old ones. Whatever regularities and organization there is in these new patterns could then become the new terms for describing the world.

2 The Binary Spatter Code

The spatter code is a particularly simple way to construct new patterns from combinations of old ones. It does so in a way that would be natural for neurons, and it could generate endless raw material for the building of new, high-level terms. The binary spatter code, which is described here, is the simplest. The idea generalizes readily to nonbinary, and some things that are problematic for the binary code can be avoided.

It has been shown in psychological experiments that our short-term memory, or span of immediate attention, or mental focus has a limited capacity, and that the capacity is best expressed in chunks. The number of chunks is around seven (Miller, 1956), whereas what constitutes a chunk is very flexible. When chunks are made of chunks, we can deal with things in the world both generally and in great detail. The low-level chunks then represent concrete things and detail, and the high-level chunks represent abstract things and the general (Albus, 1991).

The great freedom in defining and combining chunks suggests that chunks of all kinds and at all levels are represented internally in the same way. The representation is neutral as to what it represents. The binary spatter code takes this idea into the extreme: chunks are represented by large, random bit patterns. The main conditions for these patterns—that is, for the codewords for the chunks—are that

1. as several chunks combine to form a new chunk, the codewords for the components should be visible in the codeword for the new chunk (the new codeword should be *analyzable*); and
2. the new chunk should be represented by a codeword that can be used in the construction of codewords for further chunks (the code should be *recursive*).

To satisfy both conditions, the spatter code represents each chunk by two related codewords: a *dense* codeword, to satisfy condition 1, and a *sparse* codeword, to satisfy condition 2. If the code space is large enough (e.g., 10,000-bit code), the sparse codewords by themselves are both analyzable and recursive.

```
      bit 1                                   bit 10,000
      /                                       /
w1    0 0 0 0 0 1 0 0 0 0 0 0 0 0 … 1 … 0 0 0
w2    0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 0 … 0 0 1
w3    0 0 0 0 0 1 0 0 0 0 0 0 1 0 … 1 … 0 0 0
w4    0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 … 0 … 0 0 00.058  1s
w5    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 0 … 0 0 0
w6    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 … 1 … 0 0 0
w7    0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 … 0 … 0 0 0
-------------------------------------------------------------
Sum   0 1 0 0 0 2 0 0 0 0 0 1 1 0 … 3 … 0 0 1
W8    0 1 0 0 0 1 0 0 0 0 0 1 1 0 … 1 … 0 0 10.34  1s
w8    0 0 0 0 0 1 0 0 0 0 0 0 0 0 … 1 … 0 0 00.058  1s
```

Figure 1. Generating dense and sparse codewords **W**8 and **w**8 from the sparse codewords of seven constituents, **w**1, **w**2, …, **w**7. If the probability of a 1 in the sparse words is 0.058, the code will be recursive and the probability of a 1 in the dense words will be 0.34.

Table 1

The Probability of a 1 in the Sparse (p) and Dense (P) Codewords of a Recursive Spatter Code that Combines n Chunks

n	p	P
3	0.5	0.875
5	0.131	0.504
7	0.058	0.342
9	0.032	0.256
11	0.021	0.204

Figure 1 shows the construction of a (10,000-bit) binary spatter code when seven old chunks are combined into a new chunk. The dense codeword for the new chunk is simply the logical OR of the sparse codewords of the seven constituent chunks. The sparse codeword for the new chunk is built from overlaps: it has 1s where two or more constituent codewords have a 1.

In a recursive code, the probability p of a 1 in the sparse words depends on the number n of chunks that are combined into a new chunk. The recursive p is gotten by solving

$$q^n + npq^{n-1} = q$$

for p, where $q = 1 - p$ (assuming that the constituents are random and independent). The left side of the equation expresses the probability that a bit of the new sparse word will be a 0 (i.e., when all n constituent words have a 0, or when any one of them has a 1 and the rest have a 0). This probability should be the same (q) as it is in the constituent words. The dense codewords will then have 1s with probability $P = 1 \quad q^n$. Table 1 gives the recursive probabilities for several chunking factors.

3 Discussion

The neural character of the spatter code is evident when the two new codewords are construed as linear threshold functions of their constituents: the (vector) sum of the constituent codewords is thresholded anywhere between 0 and 1 (e.g., at 0.5) to get the dense word, and it is thresholded anywhere between 1 and 2 (e.g., at 1.5) to get the sparse word (see Fig. 1). Furthermore, the sparseness of the sparse words agrees with the low level of activity of many neural circuits, and large patterns and randomness agree with the size and connectivity of many neural circuits.

The code is strongly directional. Given the component codewords, the two new codewords fall out most naturally. However, a codeword's constituents cannot be determined from the codeword, although it is easy to decide (with logical AND) whether one codeword is a constituent of another, and in that sense the code is ana-

lyzable. The synthetic and holistic nature of the human mind could be related to such issues of directionality and analyzability.

3.1 Relation to other work

This work was influenced most strongly by the scatter code of Smith and Stanford (1990; Stanford & Smith, 1993 ms.) that uses randomness and logical XOR to map real variables into Hamming space. The other major inspiration was Manevitz and Zemach's (1991) paper on multilevel information processing in an associative memory, which encodes *sequence* data at multiple levels; here, the elements or chunks that go into making higher level chunks are an *unordered* set. Jaeckel's (1989) encoding of simple visual images considers unordered sets of image components and constructs the encoding of the image with the logical OR of very long bit strings that encode image components.

3.2 Acknowledgment

I am most grateful to Derek Smith for his visit with me at the Santa Fe Institute in February of 1993, to tell me about the Scatter Code. I chose the term Spatter Code partly to acknowledge its close kinship to the Scatter Code, and partly to capture the ideas of sparseness, randomness, and hitting some spots more than once when you spatter several (7) times over the same area.

References

Albus, J.S. (1991) Outline for a theory of intelligence. *IEEE Trans. Systems, Man, and Cybernetics* **31**(3):473–509.

Hassoun, M.H., ed. (1993) *Associative Neural Memories: Theory and Implementation*. New York: Oxford University Press.

Jaeckel, L.A. (1989) Some Methods of Encoding Simple Visual Images for Use with a Sparse Distributed Memory, with Application to Character Recognition. Report RIACS TR 89.29, Research Institute for Advanced Computer Science, NASA–Ames Research Center.

Kohonen, T. (1988) *Self-Organization and Associative Memory,* 2nd ed. Berlin: Springer–Verlag.

Manevitz, L.M., and Zemach, Y. (1991) Assigning meaning to data: Multi-level information processing in Kanerva's SDM. *Proc. 1991 Israel Conference on AI and Computer Vision (IAICV 8);* 114–130.

Miller, G.A. (1956) The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review* **63**:71–97.

Smith, D.J., and Stanford, P.H. (1990) A random walk in Hamming space. *Proc. 1990 Int'l Joint Conference on Neural Networks (IJCNN 90),* vol. 2; 465–470.

Stanford, P.H., and Smith, D.J. (1993 ms.) The Multidimensional Scatter Code: A Data Fusion Technique with Exponential Capacity. Submitted to ICANN 94.

Willshaw, D.J., and von der Malsburg, Ch. (1975) How patterned neural connexions can be set up by self-organisation. *Proc. Royal Society B.,* **194**:431–445.

Willshaw, D.J., and von der Malsburg, Ch. (1979) A marker induction mechanism for the establishment of ordered neural mappings: its application to the retinotectal problem. *Phil. Trans. Royal Society B.,* **287**:203–243.