A RANDOM WALK IN HAMMING SPACE

Derek Smith and Paul Stanford

Texas Instruments, Dallas, TX, 75265

Abstract

In this paper we describe the scatter code, a new technique for mapping sensor data into a form suitable for processing by associative neural networks. By overthrowing the assumption that the sensor mapping must preserve order in addition to closeness, a mapping can be generated that has much more capacity and flexibility than previous mappings. The first advantage is the practically limitless number of points in the new mapping, which eliminates the need to compress sensor information and lose resolution. The second advantage is the ability to control the radius of association in the mapped sensor data by varying the rate at which points in the hamming space become orthogonal. The radius of association can be varied within the mapping to take advantage of special cases when information about the sensor or about the expected data points is known.

Introduction

Some form of mapping between sensor data and associative neural networks [Kanerva88, Albus81, Hopfield82] is required because sensor data typically forms a range of real numbers or integers, whereas associative neural networks typically work in the equivalent of hamming space. Simple mappings from linear space into hamming space, such as binary encodings, fail because they do not preserve closeness in hamming space. Codes such as the thermometer [Hancock88] and bar codes [Penz87] preserve closeness, but can only map a restricted number of points. The scatter code preserves closeness and can map a very large number of points.

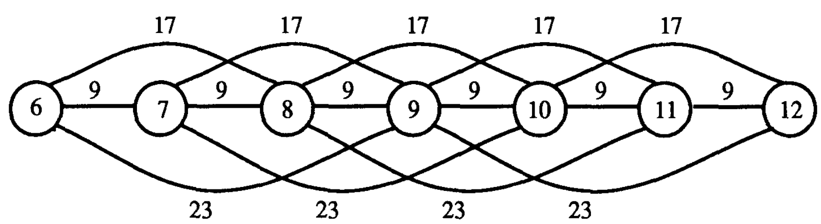
Terminology

We use n for the number of bits in the hamming space, b for the number of bits to flip at each step in the scatter code, and s for the proportion b/n. Two points are *close* or *associate* if their hamming distance is less than about $n/4$. Two points are effectively *orthogonal* if their hamming distance is about $n/2$; most randomly selected points are orthogonal [Kanerva88]. For a given point, only those points within its *radius of association* can associate.

How the scatter code is generated

A sequence of codes to correspond with a range of points in the linear space is generated as follows. The first code is chosen randomly. The second code is generated from the first by randomly choosing b bit locations (not necessarily distinct) and flipping the bit in that location, changing a zero to a one, or a one to a zero. The third code is generated from the second in the same way as the second was generated from the first, and so on for all subsequent codes until the sequence is long enough to cover the range of the linear space.

Figure 1: This figure illustrates the expected hamming distance between pairs of circled sensor data points, mapped using s=0.1 and $n=100$.

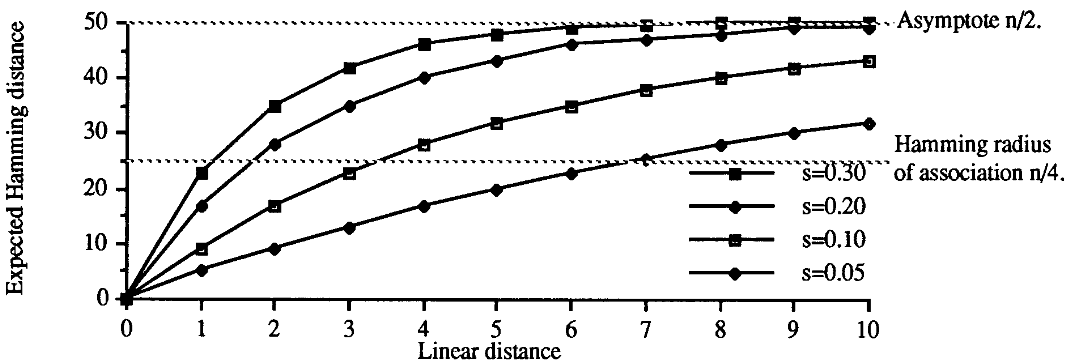


If the neural network for which the code is to be generated associates points that are within $n/4$ of each other, then with the above encoding points within 3 in the linear space will associate. Reducing s to 0.05 causes points within 7 in the linear space to associate. Thus by varying s the radius of association in the linear space can be controlled.

Expected behavior of the scatter code

The formula *n/2 * (1 - (1 - 2/n)**(s*t))* gives the expected hamming distance for points *t* apart in the linear space. The expected hamming distance asymptotically approaches *n/2* bits, where the probability of flipping a 1 to a 0 is equal to that of flipping a 0 to a 1.

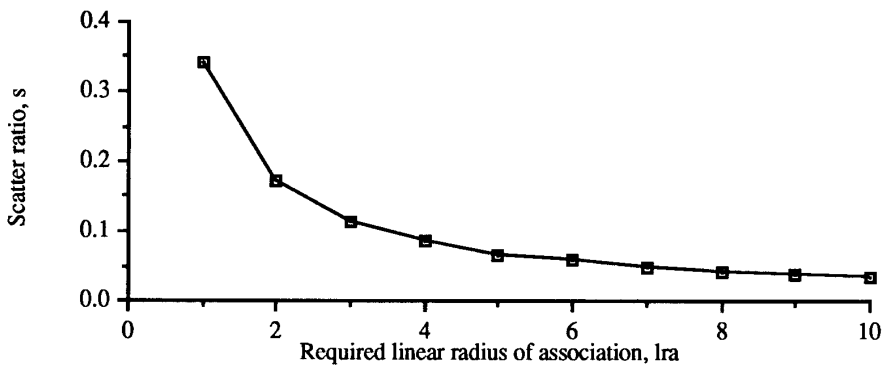
Figure 2: This graph illustrates that by varying s, the linear radius of association can be varied.



The formula *log(1 - 2*r/n) / log(1 - 2/n)* gives the expected number of bit flips it takes to be hamming distance *r* from the origin. The required linear radius of association, *lra*, of the sensor data can be achieved for a given hamming radius of association, *hra*, of the associative memory, by selecting *s* according to the following equation.

$$s = \{log(1 - 2hra/n) / log(1 - 2/n)\}/(n.lra)$$

Figure 3: This graph shows the relationship between the desired linear radius of association and the scatter ratio.



Variance of the scatter code

Figure 4: Because scatter codes are randomly generated the above equations only give expected values. The following table shows expected hamming distance, and actual hamming distance for n=100 s=0.1.

Integer distance	0	1	2	3	4	5	6	7	8	9	10
Expected hamming distance	0	9	17	23	28	32	35	38	40	42	43
Actual hamming distance	0	9	15	23	27	31	35	39	39	41	41

The variance of the scatter code is given by the equation:

$$\text{variance}^2 = \sum_{r=0}^{f} p(f,r).(r\text{-}ed(f))^2$$

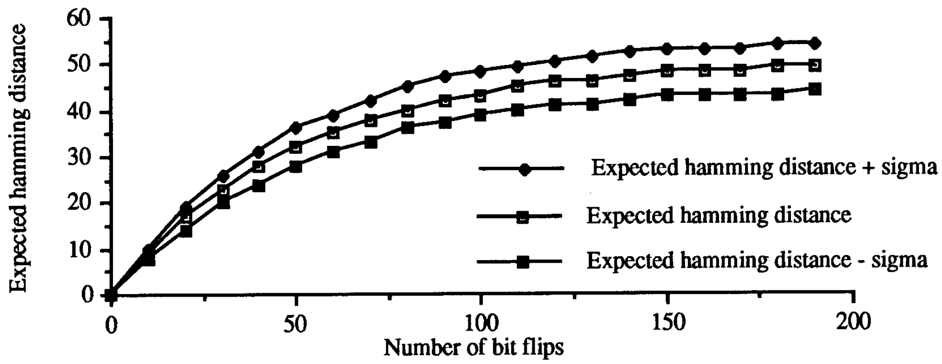
where *p(f,r)* is the probability of being r from the origin after *f* bit flips, given by:

$$p(0,0) = 1; \quad p(0,r+1) = 0;$$
$$p(f+1,r) = p(f,r-1).(n-r+1)/n \; + \; p(f,r+1).(r+1)/n$$

and *ed(f)* is the expected hamming distance from the origin after *f* bit flips, given by:

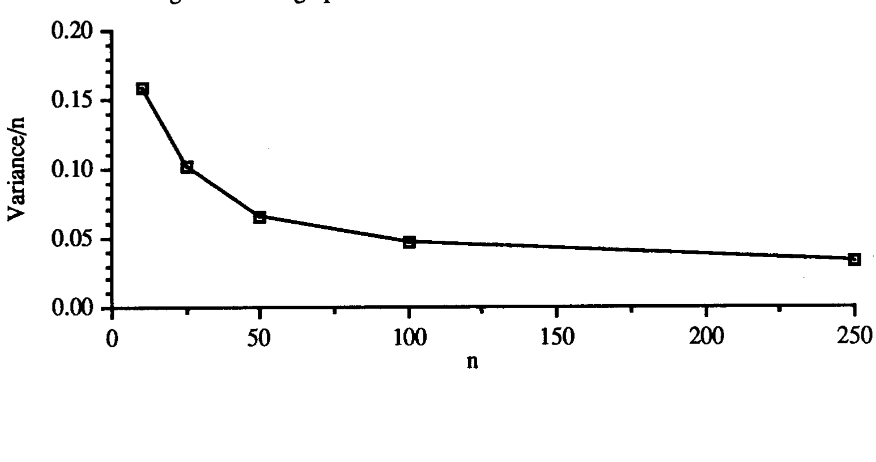
$$\sum_{r=0}^{f} p(f,r) \,.\, r \;=\; n/2 \,.\, (1 - (1 - 2/n)^f)$$

Figure 5: This graph shows sigma bounds on the expected hamming distance (n=100).



The following graph shows that the variance is quite high when *n* is less than 50. Its use on such short codes will only be useful if the problem can withstand this variance. At 100 bits the variance is better and at 250 bits better still. We typically use codes of 100 or 256 bits length, on a sparse distributed memory [Kanerva88].

Figure 6: This graph illustrates the ratio of variance to n, as n varies.



Capacity

The limit to capacity for a scatter code is the risk that widely separate linear points will map to close hamming points. If *risk* denotes an acceptable probability of scatter collision and the hamming threshold is *h.n* then the capacity is approximately:

$$(2h^h)^{n/2}.\sqrt{(\mathrm{risk}/n)}$$

A typical value of h is approximately 0.25 for sparse distributed memory [Kanerva88], giving a capacity of approximately:

$$2^{n/4}.\sqrt{(\mathrm{risk}/n)}$$

Figure 7: This table gives the estimated capacity of the scatter encoding for typical values of *risk* and *n*.

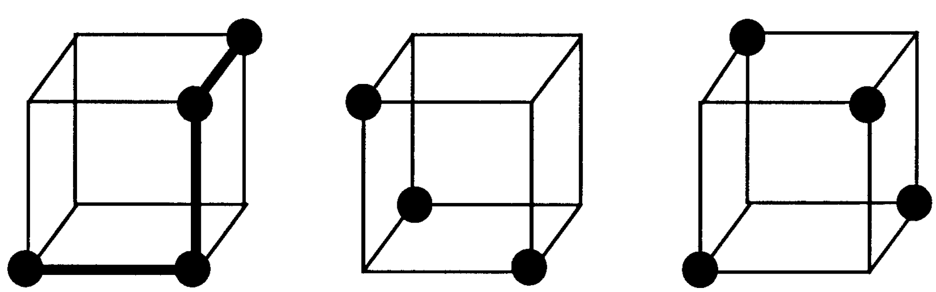
n	100	200	256	256
Risk	10^{-6}	10^{-9}	10^{-9}	10^{-12}
Capacity	17,000	10^{10}	10^{14}	10^{12}

The number of bits in a binary representation with the same capacity as a scatter code with *risk* 2^{-p} is approximately $n/4 - p/2$.[1]

Comparison with other codes

For purposes of unification, b, and thus s, can be can be given an interpretation for the thermometer and bar codes as well as the scatter code. For the thermometer code b bits are flipped on each step. However they are not chosen at random: they are always bits that have not previously been flipped. Similarly for the bar code b bits are flipped on each step. Again the bits are not chosen at random: $b/2$ are bits that were previously flipped, and $b/2$ are bits that were not flipped.[2]

Figure 8: This is a pictorial representation of the thermometer, bar and scatter codes in the (unrealistic, but drawable) case of n=3.



Thermometer code (*b=1*)
0 <> 000
1 <> 001
2 <> 011
3 <> 111

Bar code (*b=2*)
0 <> 001
1 <> 010
2 <> 100

Example Scatter code (*b=2*)
0 <> 000
1 <> 011
2 <> 110
3 <> 101

1. An efficient mapping is important in cases where the associative memory has exponential storage capacity [Chou87].
2. A middle ground between the scatter and bar codes could be a scatter code that keeps the number of ones constant. Such a code would have less capacity than the unrestricted scatter code but could be used to generate the sparse codes which are preferred by some associative memories [Marr69].

The thermometer code orders points along a *chain* from 000 to 111. The bar code orders points around a *circle* centered at the origin, radius one, in this case. The scatter code is not restricted to preserving order and *scatters* successive points moderate distances in random directions.

No need for compression

In the case of the thermometer code the maximum number of distinct codes is equal to $n+1$, the number of bits in the code plus one. In the bar code the number of codes is equal to n, the number of bits in the code.[3] In the scatter code the number of possible codes is exponential in the number of bits in the code, approximately $(2^{**}(n/4)).\sqrt{(risk/n)}$. This gives the scatter code the huge advantage that data does not have to be compressed to fit into the restricted number of codes available, avoiding the loss of resolution caused by such a compression.

For example, in the actual data below there are integers ranging from 40 to 700. If this data is to be mapped into a 100 dimensional hamming space then previous codes would restrict the mapping to 100 points. Thus the data must be compressed into the range 0 to 99 as shown below.

Figure 9: This table shows an example of loss of resolution caused by data compression.

Actual data	40	41	43	140	143	170	220	700
Compressed data	0	0	1	16	17	20	28	99

This compression has disastrous results. Because of the extreme point 700, the rest of the data (between 40 and 220) is squashed together resulting in loss of resolution. Compression is not necessary for the scatter code because of its large capacity.

Randomness

With the non-random thermometer and bar codes the hamming distances are exact and a simple algorithm can be used to map between the linear space and the hamming space. Because the scatter code is randomly generated, hamming distances are probabilistic and look-up tables are required for the encode and decode operations. The size of the tables grows linearly with the number of points encoded.

Closeness, not order and closeness

Thermometer and bar codes preserve the ordering of the linear space in the hamming space. This order is superfluous as ordering is irrelevant for hamming distance. The scatter code preserves only closeness, and it is this relaxation that is exploited in the scatter code to produce an exponentially more efficient encoding.

The special case of knowing the expected data

If the problem to be solved were the classification of noisy sensor information, and it was known that the accuracy of the sensor was not constant over the range of operation, then modifications to the scatter code could take this into account. As the scatter code is generated sequentially, the scatter at each step does not have to be constant. Because increasing the scatter decreases the radius of association, the code could be tuned to have a larger radius of association in regions of low resolution in the sensor and low radius of association in the regions of high resolution.

Similarly, if it is known that the data in one part of the range comprises many data points with little noise, then the code for that range can have small linear step size and a low radius of association. If in other parts of the range there are few data points but a lot of noise, then the linear step size can be increased, and the radius of association will also be large.

3. A technique can be used to increase the capacity of the thermometer and bar codes to $(n^{**}2)/4$ [Penz87]. However, by introducing exponential significance into the mapping this technique requires a more complex metric to be used in place of hamming distance in the underlying associative memory. The scatter code achieves its high capacity without introducing exponential significance.

There is no need to generate the code in a range which is known to be devoid of interesting data. Restarting the code from a random point at the next interesting part of the linear space is valid, because the result of several (unused) steps in the random walk is effectively independent of its starting point. A code can be fragmented many times in this way; the only consequence will be the reduction of look-up table size for mapping between the linear and hamming spaces.

Summary

Simple mappings from linear space into hamming space, such as binary encodings, fail because they do not preserve closeness in hamming space. Codes such as the thermometer and bar codes preserve closeness, but can only map a restricted number of points. By overthrowing the assumption that the sensor mapping must preserve order in addition to closeness, a mapping can be generated that has much more capacity and flexibility than previous mappings. The first advantage is the practically limitless number of points in the new mapping, which eliminates the need to compress sensor information and lose resolution. The second advantage is the ability to control the radius of association in the mapped sensor data by varying the rate at which points in the hamming space become orthogonal. The radius of association can be varied within the mapping to take advantage of special cases when information about the sensor or about the expected data points is known.

References

[Albus81] Albus,J.S., *Brains, Behavior, and Robotics*, Byte Books, 1981.

[Chou87] Chou,P., *The Capacity of the Kanerva Associative Memory is Exponential*, in proc. IEEE conf. Neural Information processing, 1987.

[Hancock88] Hancock,P.J.B., *Data representation in neural nets: an empirical study*, in proc. Connectionist Summer School, 1988.

[Hopfield82] Hopfield,J.J., *Neural networks and physical systems with emergent collective computational capabilities*, in proc. National Academy of Sciences (Biophysics) 79(8):2554-2558, 1982.

[Kanerva88] Kanerva,P., *Sparse Distributed Memory*, MIT Press, 1988.

[Marr69] Marr,D., *A Theory of the Cerebellar Cortex*, in Journal of Physiology 202:437-470, 1969.

[Penz87] Penz,P.A.. *The Closeness code: An Integer to Binary Vector Transformation Suitable For Neural Network Algorithms*, in proc. IEEE First International Conference on Neural Networks, 1987.