RIACS

Research Institute for Advanced Computer Science
NASA Ames Research Center

KANERVA'S SPARSE DISTRIBUTED MEMORY WITH MULTIPLE HAMMING THRESHOLDS

P-9

SEPPO POHJA

and

KIMMO KASKI

RIACS Technical Report 92.06

March 1992

RIACS

TECHNICAL REPORT 92.06

MARCH 1992

KANERVA'S SPARSE DISTRIBUTED MEMORY WITH MULTIPLE HAMMING THRESHOLDS

SEPPO POHJA
and
KIMMO KASKI

RIACS

Technical Report 92.06

March 1992

KANERVA'S SPARSE DISTRIBUTED MEMORY WITH MULTIPLE HAMMING THRESHOLDS

by

SEPPO POHJA and KIMMO KASKI

ABSTRACT

If the stored input patterns of Kanerva's Sparse Distributed Memory (SDM) are highly correlated, utilization of the storage capacity is very low compared to the case of uniformly distributed random input patterns. Here we consider a variation of SDM that has a better storage capacity utilization for correlated input patterns. This approach uses a separate selection threshold for each physical storage address or hard location. The selection of the hard locations for reading or writing can be done in parallel of which SDM implementations can benefit.

KANERVA'S SPARSE DISTRIBUTED MEMORY WITH MULTIPLE HAMMING THRESHOLDS

Seppo Pohja[1,2] and Kimmo Kaski[1]

1.Tampere University of Technology

Microelectronics Laboratory

2. Stanford University

Electrical Engineering Department

and

Research Institute for Advanced Computer Science

NASA-Ames Research Center

1. INTRODUCTION

The original Sparse Distributed Memory developed by Pentti Kanerva [2] is best suited for handling data with uniform random distribution of bits. Also, its physical storage addresses or hard locations are distributed in the same fashion. Natural, real-world data patterns are hardly ever like this. Rather, they come in clusters, which leaves large portions of the memory under-utilized while other parts are overwhelmed with data. In these overutilized portions the stored patterns interfere with each other as they share most or all of each other's storage locations. This causes SDM to have a much lower storage capacity utilization for natural data than uniformly distributed random data. The purpose of this study is to present and discuss a method that can be used to enhance the utilization of the capacity.

2. SPARSE DISTRIBUTED MEMORY

Sparse distributed memory can be visualized as a three-layer feed-forward network. Its structure and operation can, however, be well understood by comparing it with a conventional random-access memory or RAM system. Whereas a typical RAM system has a small address space (2^{16} or 2^{32}), the address space of SDM can be virtually any size, for instance 2^{10000}, which would not be realizable by any RAM system. In SDM the address space is covered sparsely, unlike in RAM. This means that only some of the possible address locations are physically present, the so-called hard locations, and each one of them can be activated by more than one input address during a read or write operation. SDM is also a distributed memory, as more than one hard location will be activated by any given input address.

The internal structure of SDM is shown in Figure 1. There are two matrices A and C. The matrix A is used to store addresses of the hard locations in its rows. The hard locations are initialized to be uniformly distributed random vectors of 0s and 1s. The matrix C is an array of counters initialized to zeros. They are used to store what is written onto SDM. Whenever the memory is accessed by a given address (address input vector **x**), a Hamming distance or the number of differing bits is computed for each hard location. If the Hamming distance is less than or equal to a predetermined

threshold, the bit in the activation vector **a** corresponding to hard location is set to 1 to mark an activated row. Otherwise it is set to 0 and the row will be inactive.

When writing a data input vector **i** into the memory, there are three options for each position in matrix **C**. If the row is inactive, nothing is done. Otherwise, the counter in this position is incremented when the bit in **i** corresponding to the column is 1 and decremented when the bit is 0.

When reading the memory, a sum vector is formed. Each value in the sum vector **s** is obtained by adding together all the counters in the corresponding column, provided that the row of the counter has been activated. The final output for each column is 1 if the sum is greater than 0. Otherwise, the output is 0.

Let us now consider how to overcome the problem discussed in the introduction. One possibility is to modify the address matrix **A** to fit the training set or data set. Rogers [4] has used genetic algorithms while experimenting with weather data and Saarinen et al. [5] have used Kohonen's self-organization algorithm for the modification. Joglekar [1] has taken patterns from the data itself to be used as hard-location addresses. A different approach is used by Kanerva [3] as he selectively weights the input bits while computing the Hamming distance. Many other possibilities exist. For example, Vanhala [6] has studied randomizing input to make it spread more evenly.

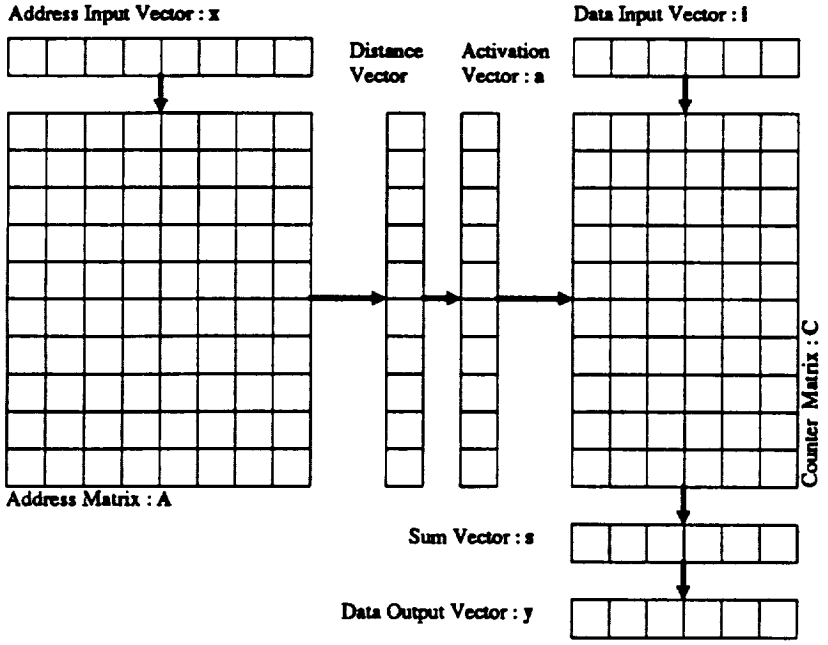

Figure 1. Structure of sparse distributed memory.

3. NEW TRAINING ALGORITHM FOR SDM

Here we introduce a new training algorithm in which a separate threshold is set for each of the hard locations as opposed to the original global threshold that is common to all the hard locations. This makes training a two-step process in which we start by finding the threshold values and, after that, continue by writing to the memory in the way we would do with the original SDM algorithm.

In the first stage we need a criterion for choosing the threshold values. If we used the original SDM, we would choose some propability P for a randomly chosen hard location to be activated. A good rule of thumb is to choose P equal to the ratio of square root of m and m itself, where m is the total number of hard locations. The global threshold could be calculated independently of the training set. It would give us in average at least t times P activations per hard location, where t is the size of the training set. However, when the data used are highly correlated, most hard locations become activated either very often or very seldom. We would like them to be activated approximately equally often and therefore we state our new criterion as follows:

1. *The threshold value for each hard location is the smallest value that yields a propability* P *or higher of the hard location's being activated.*

It should be noted that P has the same value for all the hard locations. Since the criterion is not very good for hardware implementations, we restate it in terms of a training set of t items.

2. *The threshold value for each hard location is the smallest value that results in at least* t *times* P *activations of that location by the training set.*

The second stage is identical to the training of the original SDM, with the exception that we use the previously determined treshholds for the activation process instead of a global threshold. The same note on thresholds applies to reading from the memory, as well.

4. DISCUSSION ON ALGORITHM

As we compare this algorithm with the so-called area-based selection, which is used in some of the methods that modify the hard-location addresses, we get a better understanding of the way the new algorithm works. The area-based selection method tells us to find the k closest matches or the k rows with the smallest Hamming distances to the input address. The final activation threshold is the largest Hamming distance within the found group. Finally, all the rows with Hamming distance smaller than or equal to the threshold are activated or selected. This yields $k+$ activated rows, where $k+$ is greater than or equal to k.

First, the influence of the hard-location modification is to make the distribution of the hard locations to resemble very closely that of the training set. As the area-based selection is used after the modification, the final activation threshold is the smaller the denser the hard-location distribution in the neighbourhood of the activation address. Therefore the threshold is the smaller the denser the distribution of the training set, too. Similar to this activation threshold, the multiple thresholds in the new algorithm are the smaller the denser the distribution of the training set. However, the distribution of the hard-locations does not play a role in the process of finding the thresholds.

Therefore using the new algorithm with the present hard-location modification methods is not necessarily an overly fruitfull idea, and further work needs to be done to determine which factors should govern the choise of the hard location distribution when the new algorithm is used.

In hardware realizations, the new algorithm can be made more parallel than area-based selection since the activation process happens in each row independently of the other rows. Also, the first stage can be done in parallel.

From the biological point of view, a two-stage training algorithm does not sound very plausible. However, the idea of the stages happening simultaneously during the growth of a creature sounds more appealing. In this case the first stage would dominate the beginning of the life cycle and would diminish in influence very rapidly. The second stage of the new algorithm, on the other hand, sounds plausible if the first stage has already happened either during the evolution of the species or in the past of the biological creature.

Even though not explored further, the idea of using multiple thresholds in this way for address decoding was originally introduced by Kanerva [2].

5. EXPERIMENTAL RESULTS

In these experiments we used a set of 26 5-by-7-bit patterns representing the English capital letters. It is the same set used by Kanerva [3]. In addition, we used uniformly distributed 35-bit random sets of the same size for comparison. The training was done autoassociatively and therefore the address and data parts of a training pair were identical.The memory had 300 hard locations.

We also tested a further variation of the new algorithm. For the first training stage, we created new data. This new data set included ten copies of each of the original patterns. However, some noise was added to the new set. Each bit had a five per cent change of getting flipped. This way the new set was ten times larger than the original one and had a very similar but, in a sense, smoother distribution than the smaller set. For the second stage, the original set was used.

There were four tests run ten times: (1) Unmodified SDM was trained with the data set; (2) SDM was trained with the data set using the new algorithm; (3) SDM was trained with the data set using the variation of the new algorithm; (4) Unmodified SDM was trained with uniformly distributed random set.

The results are summarized in Table 1. Entries of the form *x(y)z* give the smallest, the mean, and the largest of the ten values obtained.

The row labeled *Empty cells* gives the number of hard locations that were not activated at all. *Mean pats./cell* tells how many times the cells were activated in average and *Max. pats./cell* gives the number of activations of the most active cell. *Var. pats./cell* is the variance of the number of activations per hard location. *Mean % overlap* is the average overlap over all ordered pairs of addresses in the training set and *Max. % overlap* is the largest of them. The overlap tells how large portion of the hard locations activated by the first address are activated also by the second one.

Figure 2 shows how many per cent of the trained figures were retrieved correctly as a function of the number of trained patterns in each test. The percentage shown is the mean of the results of the ten runs. The patterns were used in alphabetical order for training.

6. SUMMARY

In this paper we have introduced a new algorithm for SDM training. It requires an additional element to the SDM realizations for storing thresholds, but it does not reduce the parallelism of these realizations. Our experiments support the suggestion that this algorithm improves SDM's ability to store correlated data.

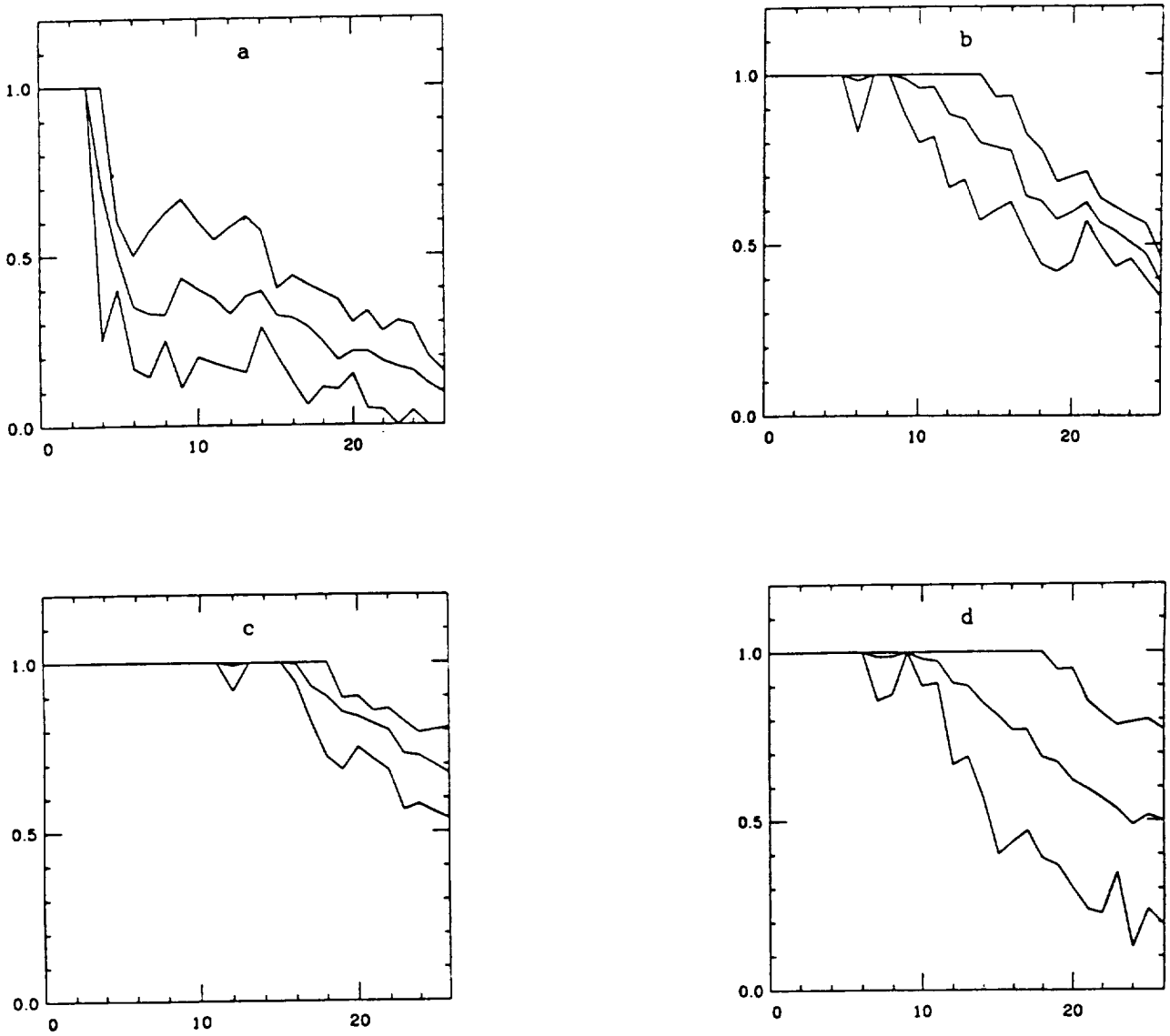

Figure 2. The minimum, maximum, and mean capacity. (a) basic SDM, (b) SDM with the new algorithm, and (c)SDM with a variation of the new algorithm, (d) SDM; random data.

Table 1

Summarized Results

	SDM	New algorithm	Variation of new algorithm	SDM - Random data
Empty cells	73 (85) 95	0 (0) 0	2 (6.7) 12	16 (26.1) 35
Mean pats./cell	2.14 (2.29) 2.55	2.71 (2.83) 2.94	2.10 (2.20) 2.28	2.18 (2.29) 2.40
Max. pats/cell	11 (13.8) 16	7 (7.7) 9	5 (5.9) 7	6 (7.7) 9
Var. pats/cell	5.91 (6.64) 7.88	1.08 (1.28) 1.60	0.86 (1.04) 1.23	1.63 (2.06) 2.48
Mean % overlap	15.5 (16.7) 18.5	8.58 (9.20) 9.86	6.43 (6.86) 7.40	7.91 (8.74) 9.61
Max.% overlap	68.0 (74.0) 87.1	51.4 (62.89) 81.6	50.0 (59.5) 73.3	34.6 (44.9) 62.5

7. REFERENCES

[1]Joglekar, U. D. (1989) Learning to Read Aloud: A Neural Network Approach Using Sparse Distributed Memory. Report No. 89.27, RIACS, NASA Ames Research Center, Moffett Field, California.

[2]Kanerva, P. (1988) *Sparse Distributed Memory*. Cambridge, Massachusetts: MIT Press.

[3]Kanerva, P. (1991) Efficient Packing of Patterns in Sparse Distributed Memory by Selective Weighting of Input Bits. *Artificial Neural Networks*, Proceedings of the 1991 International Conference on Artificial Neural Networks (ICANN-91 Espoo) 1:279-284. Amsterdam: Elsevier Science Publishers B.V.

[4]Rogers, D. (1990) Predicting weather using a genetic memory: A combination of Kanerva's sparse distributed memory and Holland's genetic algorithms. In D. S. Touretzky (ed.), *Advances in Neural Information Processing Systems 2*. San Mateo, California.: Kaufmann.

[5]Saarinen, J., Pohja, S., and Kaski, K. (1991) Self-organization with Kanerva's Sparse Distributed Memory. *Artificial Neural Networks*, Proceedings of the 1991 International Conference on Artificial Neural Networks (ICANN-91 Espoo) 1:285-290. Amsterdam: Elsevier Science Publishers B.V.

[6]Vanhala, J. (1990) Neural Networks and Distributed Computing. Thesis for the degree of Licentiate in Technology, Tampere University of Technology, Tampere, Finland.

RIACS
Mail Stop 230-5
NASA Ames Research Center
Moffett Field, CA 94035