Progressive hierarchical retrieval on associative memories

João Sacramento

Department of Computer Science and Engineering,
Technical University of Lisbon / INESC-ID, Campus Taguspark

Abstract

In this article we explore an alternative structural representation for Steinbuch-type binary associative memories. These networks offer very generous storage capacities (both asymptotic and finite) at the expense of sparse coding. However, the original retrieval prescription performs a complete search on a fully-connected network, whereas only a small fraction of units will eventually contain desired results due to the sparse coding requirement. Instead of modelling the network as a single layer of neurons we suggest a hierarchical organization where the information content of each memory is a successive approximation of one another. With such a structure it is possible to enhance retrieval performance using a progressively deepening procedure. Collected experimental evidence backing our intuition is presented, together with comments on eventual biological plausibility.

Key words: Associative memory, Steinbuch model, structural representation, hierarchical neural network, sparse coding

1. Introduction

Recent advances on the study of binary Steinbuch-type associative memories (Steinbuch, 1961) have shed light on the importance of their structural representation concerning achievable network performance (Knoblauch et al., 2009). For instance, as a means to enhance storage capacity, an alternative representation scheme based on Huffman or Goulomb coding has been proposed (Knoblauch, 2003). Instead of encoding synaptic connectivity through its immediate weight matrix form, it has been shown that if a compressed variant is used, the classic asymptotic upper bound on storage capacity may be lifted.

In this letter, we consider a novel hierarchical approach to associative memory structural representation with positive implications on retrieval performance. Furthermore, this procedure displays possible links to current neurobiology theories, being in accordance with established low energy consumption requirements (Laughlin, 2001, Lennie, 2003).

Whereas in conventional Steinbuch-type memories information is stored within a single neural network, in our method it is spread across an ensemble of hierarchically disposed networks of increased resolution. These additional networks are successive approximated (or "compressed") versions of each other and allow for early selective filtering of relevant neurons, pruning unnecessary units from the query process while progressively reaching a final result.

Hierarchical memories are already a well-known concept in neural network architecture theory but have been employed for different purposes. A trend of research has been using such structures (Kakeya and Kindo, 1996, Hirahara et al., 2000, Štanclová and Zavoral, 2005) to store

naturally correlated patterns, which would otherwise quickly saturate the network and introduce untolerable errors in the retrieval process.

2. Binary associative memories

Steinbuch-type associative memories have been subject of exhaustive analyses since their inception in the early sixties. Willshaw et al. (1969) and Palm (1980) provided a missing rigorous mathematical model description and the first formal studies on their performance. Instead of the original electrical circuit formulation envisioned by Steinbuch, they have presented the model as a single-layer feedforward network of binary threshold neurons.

These memories establish a mapping between pairs of pattern vectors $\{(x^\mu \mapsto y^\mu) : \mu = 1, 2, ..., M)\}$, where we denote $x \in \{0,1\}^m$ as the *address* or *input* vector and $y \in \{0,1\}^n$ as the *content* or *output* vector. When presented with a possibly noisy or incomplete input vector $\tilde{x}$, the network should determine the most similar stored x^μ and return the corresponding y^μ, according to some established similarity measure such as the Hamming distance. This general mapping task is referred to as *hetero-association*. If the set of stored vector pairs is of the form (x, x), the network is said to perform an *auto-association* task, useful when pattern completion is desired.

The set of M associations is learnt through a special non-linear rule, which forms the $m \times n$ (or $n \times n$ for the auto-associative case) binary synaptic weight matrix W according to:

$$W_{ij} = \min\Big(1, \sum_{\mu=1}^{M} x_i^\mu y_j^\mu\Big), \tag{1}$$

where each column W_j corresponds to the weight vector of neuron j.

Equation 1 is a simple realization of the hypothesis due to Hebb (1949), which stated that simultaneous pre- and post-synaptic activity would increase the efficacy of that synapse. The implemented variant of Hebbian learning is said to be *clipped* as it merely registers a binary-valued state for each synapse ("on" or "off"), *distributed* as correlated patterns will reuse the same memory locations, and *local* since every neuron can independently and in parallel update its weight vector.

After learning, stored associations can be recalled using a distorted input vector $\tilde{x}$ as a cue. The *dendritic sum* s_j is calculated for every neuron in a synchronous *one-step* fashion

$$s_j = \sum_i W_{ij}\tilde{x}_i \quad \forall j, \tag{2}$$

and compared with a *threshold* value Θ using a *Heaviside* transfer function:

$$y_j = H(s_j - \Theta). \tag{3}$$

Note that a well-chosen Θ is a crucial for high-quality retrieval; too low values will result in *add-errors* and too high values will result in *miss-errors*. We define an add-error as an additional "1"-component in the output vector y which was not present in the originally learnt y^μ. Similarly, a miss-error occurs when there is a missing "1"-component in the output vector y.

We use the ℓ_0 zero norm of a pattern x to denote its "activity level" $|x|_0$, i.e., the number of active elements. The classic threshold setting regime due to Willshaw et al. (1969), where $\Theta = |\tilde{x}|_0 = \sum_i \tilde{x}_i$, is an optimal solution when the input cue $\tilde{x}$ is possibly incomplete but not

noisy. For the general case where $\tilde{x}$ may contain additional or mispositioned "1"-components, the threshold must be chosen using an approximation strategy with the goal value set at $\Theta \approx \sum_i \tilde{x} x^\mu$, which may or may not be easy to accomplish as it is not a simple function of $\tilde{x}$ (Graham and Willshaw, 1995).

Several variations on the original retrieval process have been proposed (see Sommer and Palm (1999) for a careful analysis) yielding greater error tolerance at the expense of additional computational costs. We will not consider them in this work, as we aim for the lowest possible retrieval effort.

We will not deal with storage capacity and information efficiency in this work, as they have received extensive treatment in the literature (Willshaw et al., 1969, Palm, 1980, Amari, 1989, Nadal and Toulouse, 1990, Palm and Sommer, 1996, Knoblauch et al., 2009). However, for clarity's sake, let us refer that to achieve optimal behaviour a Steinbuch-type memory should be loaded with associations of uncorrelated sparse vectors, where their activity level is of logarithmic order of their length. If this constraint is met, Steinbuch-type memories will display unrivalled practical information capacities of $C = \ln 2$, much greater than those of Hopfield-type networks whose asymptotic upper bound is set around $C^H = 1/(4 \ln 2)$, with practical expected values somewhere between $C^H/2$ and C^H (Palm and Sommer, 1992).

These generous storage properties combined with the model's biological relevance have ensured continuous received attention. Technical applications have succeeded to develop domain-specific coding techniques (Rehn and Sommer, 2006, Wichert, 2006) which allow for real capacities near C to be reached. Also, when sparse coding, partial connectivity and noise-tolerant thresholding strategies are combined, Steinbuch memories become very attractive from a biological viewpoint, while maintaining high computational efficiency (Graham and Willshaw, 1994). As it has been pointed out recently, these memories are still the only known way of implementing very large Hebbian cell assemblies resembling the mammalian cortical structure (Wennekers, 2009). This is in part possible due to the simple and inexpensive *one-step retrieval* algorithm, which allows for fast lookup even when running on conventional John von Neumann computers, as we will discuss on section 3.

3. On the computational complexity of retrieval

Steinbuch memories naturally benefit from specialized massively parallel hardware implementations (see for instance Palm and Palm (1991)), as each neuron may independently perform both learning and retrieval, given that the computation is synchronous. In such a computer equipped with n processors, one for each neuron, the retrieval process will be proportional in time to the number of "1" elements of the input pattern $\tilde{x}$. As the activity level $|\tilde{x}|_0$ is close to $|x|_0$ which is of logarithmic order $O(\log m)$ due to the sparseness constraint, the time complexity of a parallel retrieval is also $O(\log m)$.

On a serial computer, the results of each neuron have to be calculated sequentially, for every active element on the input vector $\tilde{x}$. Generally, the so-called "pointer representation" format is employed, where $\tilde{x}$ is represented as a $|\tilde{x}|_0$-sized vector containing the indices of its "1" components (Bentz et al., 1989), avoiding their determination every time the retrieval process occurs, which would cost m additional steps. This way, n units must perform $|\tilde{x}|_0$ comparisons and a threshold cut, resulting in

$$t = n * |\tilde{x}|_0 + n \approx n * |\tilde{x}|_0 \tag{4}$$

operations. Assuming the classic Willshaw threshold setting strategy is used, its computational cost may be neglected. This yields a *quasilinear* time complexity of $O(n \log m)$. Notice that t

is independent from the number of stored associations μ, contrarily to the traditional list-based "brute-force" solution. In this case, the input cue must be compared through a measure function χ to every other address pattern to determine the closest match, rendering a total cost of

$$t^{\text{list}} = \sum_{i=1}^{\mu} \chi(\tilde{x}, x_i) \approx \mu * (|x|_0 + |\tilde{x}|_0). \tag{5}$$

The list-based solution is however the only viable option when a sparse coding prescription is not available for a given problem domain. No general solution to the sparseness constraint has been found so far.

4. Tree-like hierarchical memories

By inspection of the Hierarchical Subspace Tree due to Wichert (2009) we were led to an interesting question: could the properties of a tree-like structure be applied to binary associative memories in order to improve retrieval performance and minimize energy consumption?

In pursuit of this premise we conceived a hierarchical structural representation suitable for the general associative memory task. The simple intuition behind our method can be grasped through the analysis of equation 4. Whenever an input cue is presented to the network, every neuron will have to perform a recall procedure. However, as the stored patterns are sparse, only very few of them will eventually possess useful information for a given query — most units will be left out during the threshold cut. As such, if a memory containing approximated versions of the stored associations performs a recall first, the retrieved result may be used as an indicator of which neurons will fire positively. Of course, the method will work especially well if the employed approximation function does not lead to false negatives, otherwise undesired miss-errors could be introduced.

By applying the same process as a recursion, introducing additional layers of memories which are successive approximations of one another, a tree-like structure is built. It is not a tree of associative memories, as there is only one memory at each level; it is rather the output of the selective lookup process at step r which resembles the nodes at the level r of a tree.

Formally, we are dealing with an ordered set of R Steinbuch-type associative memories with a fixed address pattern space dimension m but a variable number of neurons $n_1, n_2, ..., n_R$ with $n_R = n$. Thus, the full $m \times n$ uncompressed associative memory can be found at depth R.

Our compression technique differs from the one found in Knoblauch (2003) as we are interested in creating approximated versions retaining a "no false-negatives" property rather than solely maximizing space usage and information efficiency. The easiest way to achieve this goal is to apply a Boolean OR based transform, which is somehow the binary equivalent of the arithmetic mean employed in Wichert (2009).

Whenever a pair of patterns (x, y) is presented to the full memory for learning, a transformed version $(x, \zeta_r(y))$ $\forall r : 1 \leq r < R$ is also presented to the r-th memory. We define ζ_r as a family of functions $\zeta_r : \{0,1\}^n \longrightarrow \{0,1\}^{n_r}$ where the elements of $z = \zeta_r(x)$ are given by equation:

$$z_i = \bigvee_{j=i\times a_r-(a_r-1)}^{i\times a_r-1} x_j. \tag{6}$$

The dimensions $n_1 < n_2 < ... < n_R = n$ are inversely proportional to the aggregation window factors $a_1, a_2, ..., a_R$, where $a_R = 1$, and may be expressed by the recursive relation

$$n_r = \begin{cases} n_{r+1}/a_r & \text{if } 1 \leq r < R, \\ n & \text{if } r = R. \end{cases} \tag{7}$$

Notice that in practice ζ_r carries on a partition of x onto n/a_r subvectors and then aggregates each of them using an a_r-ary Boolean OR.

The retrieval process is performed when a distorted or incomplete pattern $\tilde{x}$ is presented to the memory at $r = 1$, which corresponds to the smallest and most approximated version of the hierarchy. Likewise, on the following memories ranging from $r = 2$ to $r = R$, $\tilde{x}$ is used as the recall cue. However, the dendritic sum at level $r + 1$ will only have to be calculated for a subset of neurons.

Let $y(r)$ denote the output pattern returned by the r-th memory. Note that for a given index j of a "1" component of $y(r)$ corresponds an index set Y_j with a_r elements which identify the original uncompressed units at level $r + 1$:

$$Y_j = \{j * a_r, j * a_r + 1, ..., j * a_r + a_r\}. \tag{8}$$

These $|y(r)|_0$ sets can then be merged to form the complete set $\mathcal{Y}_{r+1}$ of indices for which the dendritic sum must be calculated at level $r + 1$:

$$\mathcal{Y}_{r+1} = \bigcup_j Y_j \quad \forall j : y_j(r) = 1. \tag{9}$$

Hence, equation 2 is kept unchanged, but should be restricted to the members of $\mathcal{Y}_{r+1}$:

$$s_j(r + 1) = \sum_i W_{ij}\tilde{x}_i \quad \forall j : j \in \mathcal{Y}_{r+1}. \tag{10}$$

Moreover, the output transfer function should also be modified in order to update only the relevant positions for which the dendritic potential has been calculated:

$$y_j(r + 1) = \begin{cases} H(s_j(r + 1) - \Theta) & \text{if } j \in \mathcal{Y}_{r+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

This retrieval process is nicely illustrated through Figure 1, where a hetero-associative task is carried on by a hierarchy of two memories.

5. Experimental results

Through a series of numerical simulations on a sequential computer we have measured the effective retrieval costs associated with varying hierarchy dispositions. The experiments have been conducted on a mid-sized square associative memory with $m = n = 2000$ neurons. Without loss of generality, the memory performed an auto-associative task — similar results should be observed at equivalent (i.e., with a higher number of stored associations M) capacity loads for hetero-association.

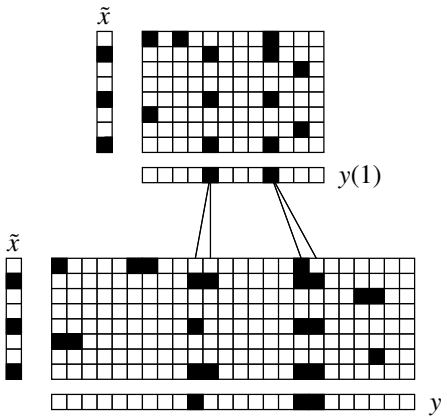

Figure 1: A tree-like hierarchical associative memory structure with $R = 2$ and $a_1 = 2$ during the retrieval process. The original synaptic weight matrix at $r = 2$ is 8×24, which yields a compressed $8 \times 24/a_1 = 12$ memory at $r = 1$. Black squares represent "1" entries.

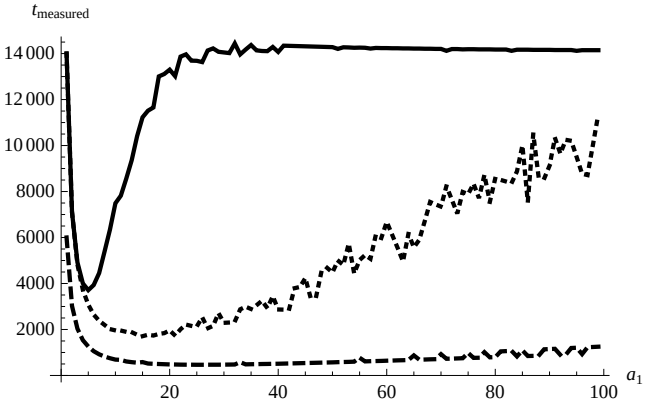

Figure 2: Measured number of computation steps (excluding threshold cuts) on a two-step aggregation ($R = 2$) hierarchy with a varying a_1 factor. The series of points were drawn from an auto-associative mid-sized memory where $n = m = 2000$ loaded with uniform random sparse patterns. The full line depicts the retrieval cost corresponding to a memory load of $M = 15000$ stored patterns with fixed logarithmic activity level $|x^\mu|_0 = 8$ where an incomplete cue with activity $|\tilde{x}|_0 = 7$ was used. Dots and dashes refer to emptier memory states where $M = 2000$, $|x^\mu|_0 = 8$ and $|\tilde{x}|_0 = 7$ (dots), $|x^\mu|_0 = 4$ and $|\tilde{x}|_0 = 3$ (dashes). On every case, $a_1 = 1$ corresponds to the classic associative memory model retrieval process, where no aggregation is performed. Notice how the costs $t^{\text{measured}} = 14000$ and $t^{\text{measured}} = 6000$ confirm equation 4.

As Fig. 2 highlights, noticeable gains may be attained with a single-step hierarchy of solely one additional memory, viz. when $R = 2$. Fuller memories tend to be more sensitive to aggregation factor choice, and an incorrect selection will easily lead to uninteresting results. This phenomenom occurs due to the successively increasing memory saturation introduced when traversing the hierarchy from $r = R$ to $r = 1$ during the learning stage. The aggregation process employed to create the smaller approximated memories will also render a capacity overload, as the resulting number of neurons will be smaller and each one will contain a higher number of active synapses. The aggregation factor should of course not be too low, as it would then deliver a sub-optimal compression, but it also should not be too high, at the expense of introducing undesired add-errors in the output patterns of the approximated memories due to excessive load.

	Minimum t^{measured} excluding threshold cuts					
Memory	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$
A	14000	3710	3122	3024	3066	3129
B	14000	1708	1071	973	917	931
C	6000	465	222	177	168	168

Table 1: The impact of the hierarchy depth factor R on the retrieval cost. Notice that $R = 1$ corresponds to the original Steinbuch associative memory. For each subsequent value of R the minimum measured number of comparisons (again, excluding threshold cuts) is shown. To reach this result, numerical optimization was employed in order to determine the ideal $a_1, ..., a_R$ aggregation factors. All memories have $n = m = 2000$ neurons and are listed from fuller to emptier, as $M_A = 15000$, $|x_A^\mu|_0 = 8$, $M_B = 2000$, $|x_B^\mu|_0 = 8$, $M_C = 2000$ and $|x_C^\mu|_0 = 4$. In both cases, pattern completion was performed using an incomplete cue with a missing "1"-component, i.e., where $|\tilde{x}|_0 = |x^\mu|_0 - 1$.

Applying the recursive aggregation through hierarchies with a depth factor $R > 2$ has also been shown useful. Table 1 illustrates performance gains achieved when optimum aggregation factors are chosen for each level $r_1, ..., r_{R-1}$. Hierarchy heights of logarithmic order $O(\log n)$ of network size resembling the properties of a tree (such as the Subspace Tree) appear to be beneficial to the retrieval process.

	Minimum t^{measured} including threshold cuts					
Memory	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$
A	16000	6110	5914	5962	5994	6042
B	16000	3832	3412	3393	2684	3417
C	8000	2537	2383	2385	2389	2483

Table 2: Repetition of the experiments performed for Table 1, but with t^{measured} including the threshold operation count. Notice that the actual ideal depths have decreased. The effective performance gains have also decreased, especially on memory C, where the weight matrix sparseness factor had provided the ideal conditions for error-free large window aggregation. Improvements of the same order of Table 1 should however still be attainable on larger networks where the cost of thresholding becomes discardable.

Although asymptotically vanishing, the $n/a_1, n/a_2, ..., n/a_R = n$ threshold operations for each memory in the hierarchy may not be neglectable on finite size memories, especially if the number of neurons is not sufficiently large. On small- or mid-sized networks, the $O(n \log m)$ retrieval

cost factor will still remain close to the $O(n)$ threshold cost. Hence, when determining the ideal architecture of such hierarchical memories, the relation between depth and extra threshold cuts must be taken into account. An increased depth factor R implies extra threshold operations due to the introduced additional memories, and the effective total retrieval cost might actually be lower on shallower trees, as shown on Table 2.

The series of measured times t^{measured} presented in Table 1 discard the number of threshold cuts on purpose; the impact of these will be near irrelevant on large memories as the asymptotic growth of the dendritic potential operation count will outpace the threshold's one. Our experiments were performed on $n = m = 2000$ mid-sized memories where we could afford to apply numerical optimization, which would take rather too long to complete on larger networks. On these networks the hierarchical retrieval process should benefit from higher R factors, achieving better effective costs, as the extra threshold cuts will be neglectable. This fact is particularly interesting since Steinbuch-type memories present optimal capacity relations for very large values of n and m (Palm, 1980).

6. Conclusions

Synaptic activity on the mammalian brain has been related to increased energy costs which may not be sustainable by the underlying metabolic processes. Our structural representation, still in the form of an abstract mathematical model, may be a simple explanation to the biological phenomena occurring on the visual cortex. Rather different hierarchical organizations have been proposed (see Lennie (1998) for an interesting discussion), but at least there is some consensus over the benefits of hierarchical structures within the human visual system.

From a computational point of view, our model has been shown useful as a means to enhance retrieval performance. The major contributing cost factor $n*|\tilde{x}|_0$ in equation 4 can be substantially diminished thanks to the tree-like hierarchical lookup procedure.

Dependence of performance gains upon network configuration and memory load has also been observed, as optimal or near-optimal results were restricted to a subset of depth and aggregation factor combinations. Different loads will correspond to different ideal sets of parameters, and sensitivity to parameter selection should increase with memory load.

Acknowledgements

The authors wish to express their gratitude towards Jan Cederquist for his helpful comments on an early version of this manuscript.

References

Amari, S., 1989. Characteristics of sparsely encoded associative memory. Neural Networks 2 (6), 451 – 457.

Bentz, H. J., Hagstroem, M., Palm, G., 1989. Information storage and effective data retrieval in sparse matrices. Neural Networks 2 (4), 289–293.
URL http://dx.doi.org/10.1016/0893-6080(89)90038-5

Graham, B., Willshaw, D., Mar. 1995. Improving recall from an associative memory. Biological Cybernetics 72 (4), 337–346.
URL http://dx.doi.org/10.1007/BF00202789

Graham, B., Willshaw, D. J., 1994. Capacity and information efficiency of a brain-like associative net. In: NIPS. pp. 513–520.

Hebb, D. O., June 1949. The Organization of Behavior: A Neuropsychological Theory. Wiley, New York.

Hirahara, M., Oka, N., Kindo, T., 2000. A cascade associative memory model with a hierarchical memory structure. Neural Networks 13 (1), 41 – 50.
URL http://dx.doi.org/10.1016/S0893-6080(99)00083-0

Kakeya, H., Kindo, T., 1996. Hierarchical concept formation in associative memory composed of neuro-window elements. Neural Networks 9 (7), 1095 – 1098.
URL http://dx.doi.org/10.1016/0893-6080(96)00030-5

Knoblauch, A., 2003. Optimal matrix compression yields storage capacity 1 for binary willshaw associative memory. In: Proceedings of the International Conference on Artificial Neural Networks and Neural Information Processing. pp. 176–176.

Knoblauch, A., Palm, G., Sommer, F. T., 2009. Memory capacities for synaptic and structural plasticity. Neural Computation (in press).

Laughlin, S. B., 2001. Energy as a constraint on the coding and processing of sensory information. Current Opinion in Neurobiology 11 (4), 475 – 480.
URL http://dx.doi.org/10.1016/S0959-4388(00)00237-3

Lennie, P., 1998. Single units and visual cortical organization. Perception 27 (8), 889–935.
URL http://www.perceptionweb.com/abstract.cgi?id=p270889

Lennie, P., 2003. The cost of cortical computation. Current Biology 13 (6), 493 – 497.
URL http://dx.doi.org/10.1016/S0960-9822(03)00135-0

Nadal, J.-P., Toulouse, G., 1990. Information storage in sparsely coded memory nets. Network: Computation in Neural Systems 1, 61–74(14).
URL http://dx.doi.org/10.1088/0954-898X/1/1/005

Palm, G., Feb. 1980. On associative memory. Biological Cybernetics 36 (1), 19–31.
URL http://dx.doi.org/10.1007/BF00337019

Palm, G., Palm, M., 1991. Parallel associative networks: The pan-system and the bacchus-chip. In: Ramacher, U., Rckert, U., Nossek, J. (Eds.), Proceedings of the Second International Conference on Microelectronics for Neural Networks. Kyrill & Method, pp. 411–416.

Palm, G., Sommer, F., 1992. Information capacity in recurrent mcculloch-pitts networks with sparsely coded memory states. Network: Computation in Neural Systems 3, 177–186(10).
URL http://dx.doi.org/10.1088/0954-898X/3/2/006

Palm, G., Sommer, F. T., 1996. Associative data storage and retrieval in neural networks. Models of Neural Networks: Association, Generalization, and Representation (Physics of Neural Networks) 3, 79–118.

Rehn, M., Sommer, F. T., 2006. Storing and restoring visual input with collaborative rank coding and associative memory. Neurocomputing 69 (10-12), 1219 – 1223, computational Neuroscience: Trends in Research 2006.
URL http://dx.doi.org/10.1016/j.neucom.2005.12.080

Sommer, F. T., Palm, G., 1999. Improved bidirectional retrieval of sparse patterns stored by hebbian learning. Neural Networks 12 (2), 281 – 297.

Steinbuch, K., 1961. Die Lernmatrix: Automat und Mensch. Springer-Verlag.

Štanclová, J., Zavoral, F., 2005. Hierarchical associative memories: The neural network for prediction in spatial maps. In: Proceedings of the 13th International Conference on Image Analysis and Processing. Vol. 3617. Springer Berlin / Heidelberg, pp. 786–793.

Wennekers, T., Jun. 2009. On the natural hierarchical composition of cliques in cell assemblies. Cognitive Computation 1 (2), 128–138.
URL http://dx.doi.org/10.1007/s12559-008-9004-5

Wichert, A., 2006. Cell assemblies for diagnostic problem-solving. Neurocomputing 69, 810–824.

Wichert, A., 2009. Subspace tree. In: International Workshop on Content-Based Multimedia Indexing Conference Proceedings, IEEE. pp. 38–43.

Willshaw, D. J., Buneman, O. P., Longuet-Higgins, H. C., Jun. 1969. Non-holographic associative memory. Nature 222 (5197), 960–962.
URL http://dx.doi.org/10.1038/222960a0