Tree-like hierarchical associative memory structures

João Sacramento*, Andreas Wichert

INESC-ID Lisboa and Instituto Superior Técnico,
Technical University of Lisbon,
Av. Prof. Dr. Aníbal Cavaco Silva, 2744-016 Porto Salvo, Portugal

Abstract

In this letter 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. To backup our intuition we provide collected experimental evidence alongside comments on eventual biological plausibility.

Keywords: 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., 2010). 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 increased.

In this letter, we consider a novel hierarchical approach to associative memory structural representation with positive implications on retrieval performance. Furthermore, this procedure establishes possible links to current neurobiology theories, complying with the low energy consumption requirements of the brain (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 arranged 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. Such structures have been used by a trend of research (Kakeya & Kindo, 1996; Hirahara et al., 2000; Štanclová & Zavoral, 2005) to store naturally correlated patterns, which would otherwise quickly saturate the network and introduce intolerable errors in the retrieval process.

*Corresponding author. Tel.: +351 21 423 32 31; fax: +351 21 314 48 43.
Email addresses: joao.sacramento@ist.utl.pt (João Sacramento), andreas.wichert@ist.utl.pt (Andreas Wichert)

2. Binary associative memories

Steinbuch-type associative memories have been object 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 $\{(x^\mu \mapsto y^\mu) : \mu = 1, 2, \ldots, M)\}$, 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 amongst the set of stored x^μ the most similar one 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 assumes 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 the learning phase is complete, stored associations can be recalled using a distorted input vector $\tilde{x}$ as a cue. The *dendritic sum* s_j is calculated for each 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 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 & Willshaw, 1995).

Several variations on the original retrieval process have been proposed (see Sommer & 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 & Toulouse, 1990; Palm & Sommer, 1996; Knoblauch et al., 2010). 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 large practical information capacities up to $C = \ln 2 \approx 0.69$. This value is slightly lower than those of sparse Hopfield-type networks which reach the asymptotic upper bound of $C^H = 1/(2 \ln 2) \approx 0.72$ (Palm & Sommer, 1992), but at the expense of synapses with non-binary weights.

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 & 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 & Willshaw, 1994). The latter is in part possible due to the simple and inexpensive *one-step retrieval* algorithm, which allows a 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 & 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

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

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 at first t seems to be independent from the number of stored associations M; however, to maintain the retrieved vector error-free, M ought not exceed its maximal value which indeed is related to n — asymptotically we have $M_{\max} \sim mn/\log^2 n$, c.f. Palm (1980).

An immediate retrieval performance improvement on sequential computer implementations arises from the suggestion found in Bentz et al. (1989) and Knoblauch et al. (2010), where each of the m matrix rows are encoded in the 'pointer representation' format. This alternative representation avoids traversing the entire content neuron population, restricting the dendritic potential calculation to units which react to the excitatory inputs.

Since sparsely-coded patterns generate balanced weight matrices where the probability p_1 of finding an active synapse settles somewhere below 0.5 as M reaches $M_{\max}$, this retrieval scheme effectively reduces the required number of computational steps on Steinbuch memories. If silent synapses are encoded, it is also applicable to other potentiation regimes which might generate dense matrices where $p_1 \to 1$. Thus, in average, each matrix row will have either $p_1 \cdot n$ or $(1 - p_1) \cdot n$ pointers. Once again excluding the threshold cuts this results in the retrieval cost of

$$t^{\text{ptr}} = \min(1 - p_1, p_1) \cdot n \cdot |\tilde{x}|_0 \tag{5}$$

computations.

It should be noted that the growth of n (and t or t^{ptr}) with M is sub-linear, unlike 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

$$t^{\text{list}} = \sum_{\mu=1}^{M} \text{cost}\left(\chi(\tilde{x}, x^{\mu})\right) \approx M \cdot (|x|_0 + |\tilde{x}|_0) \tag{6}$$

calculations.

Unlike Steinbuch-type memories, however, the list-based solution is a 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 $n_1, n_2, \ldots, n_R$ 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 $\zeta_r : \{0,1\}^{n_{r+1}} \longrightarrow \{0,1\}^{n_r}$ where the elements of $z = \zeta_r(y)$ are given by the equation:

$$z_i = \bigvee_{j=i \cdot a_r - (a_r - 1)}^{i \cdot a_r} \zeta_{r+1}(y)_j \tag{7}$$

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

$$n_r = \begin{cases} n_{r+1}/a_r & \text{if } 1 \leq r < R, \\ n & \text{if } r = R. \end{cases} \tag{8}$$

Notice that in practice ζ_r carries out a partition of x onto $n/(a_r \cdot a_{r+1} \cdot \ldots \cdot a_R)$ sub-vectors 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 any given '1' component j of $y(r)$ corresponds to an index set Y_j with a_r elements that identify the original uncompressed units at level $r + 1$:

$$Y_j = \{j \cdot a_r, j \cdot a_r - 1, \ldots, j \cdot a_r - (a_r - 1)\}. \tag{9}$$

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 $r + 1$:

$$\mathcal{Y}_{r+1} = \bigcup_j Y_j \quad \forall j : y_j(r) = 1. \tag{10}$$

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

$$s_j(r+1) = \sum_i W_{ij}\tilde{x}_i \quad \forall j : j \in \mathcal{Y}_{r+1}. \tag{11}$$

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:

$$y_j(r+1) = \begin{cases} H(s_j(r+1) - \Theta) & \text{if } j \in \mathcal{Y}_{r+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

This retrieval process is illustrated by Figure 1, where a hetero-associative task is carried out by a hierarchy of two memories. As a final remark we note that our retrieval prescription is equally valid for pointer-based sequential computer implementations.

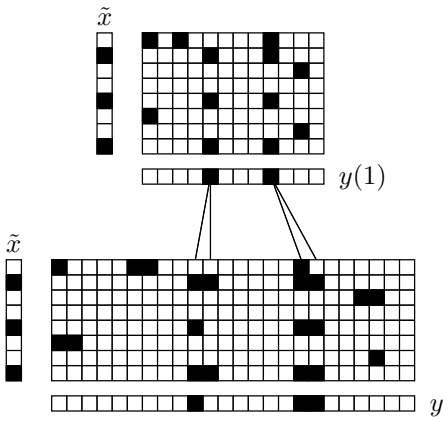

Figure 1: A tree-like hierarchical associative memory structure with $R = 2$ and $a_1 = 2$ during the retrieval process. The original synaptic weight matrix at $r = 2$ is 8×24, which yields a compressed $8 \times 24/a_1 = 12$ memory at $r = 1$. Black squares represent '1' entries.

5. Experimental results

Through a series of numerical simulations on a sequential computer we have measured the effective retrieval costs associated to 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

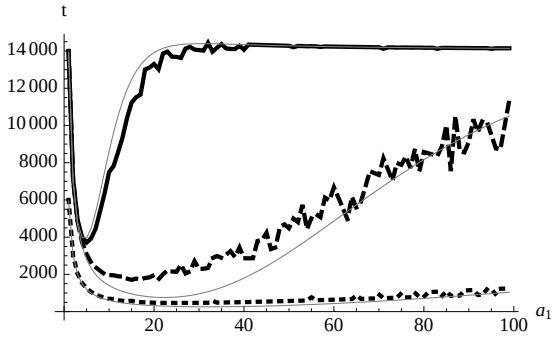

Figure 2: Thick lines indicate measured number of computation steps (excluding threshold cuts) on a two-step aggregation ($R = 2$) hierarchy with a varying a_1 factor. The series of points were drawn from an auto-associative mid-sized memory where $n = m = 2000$ loaded with uniform random sparse patterns. 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). Thin, gray lines represent analytic estimates sampled from the expressions presented on Appendix A. On every case, $a_1 = 1$ corresponds to the classic associative memory model retrieval process, where no aggregation is performed. Notice how the initial costs $t = 14000$ and $t = 6000$ confirm equation 4.

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.

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 phenomenon 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 not, of course, be too low, as it would deliver a sub-optimal compression, but it also should not be too high, at the expense of introducing an undesired level of add-errors in the output patterns of the approximated memories due to excessive load.

	Minimum t^{measured} excluding threshold cuts					
Memory	$R = 1$	$R = 2$	$R = 3$	$R = 4$	$R = 5$	$R = 6$
M_A	14000	3710	3122	3024	3066	3129
M_B	14000	1708	1071	973	917	931
M_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 $a_1, \ldots, a_R$ aggregation factors. All memories have $n = m = 2000$ neurons and are listed from fuller to emptier, as $M_A = 15000$, $|x^\mu_A|_0 = 8$, $M_B = 2000$, $|x^\mu_B|_0 = 8$, $M_C = 2000$ and $|x^\mu_C|_0 = 4$. 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 $r = 1, \ldots, r = R - 1$. 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.

The series of measured times t^{measured} presented in Table 1 discard the number of threshold cuts on purpose; these are just a constant small fraction of the total retrieval time (as illustrated by Table 2) and

	Minimum t^{measured} including threshold cuts					
Memory	$R=1$	$R=2$	$R=3$	$R=4$	$R=5$	$R=6$
M_A	16000	4240	3568	3456	3600	3616
M_B	16000	1952	1224	1112	1048	1064
M_C	8000	620	296	236	224	224

Table 2: Repetition of the experiments performed for Table 1, but with t^{measured} including the threshold operation count. The values are higher than those of Table 1 for all depth factors (including the original Steinbuch model, i.e., when $R = 1$), but the proportion of improvement is exactly the same. The only network which performs a full threshold check is the first one, which is highly compressed on a hierarchical arrangement with $R > 2$. The subsequent memories will only have to calculate the thresholds for the neurons in $\mathcal{Y}_{r+1}$, which for an optimal (or near-optimal) configuration represents a small fraction of the neural population.

their impact will be near irrelevant on large memories as the asymptotic growth of the dendritic potential operation count will outpace the threshold's.

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 become discardable. 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

Nearest neighbour determination for binary patterns is a well-known problem for which numerous approaches exist — take for instance locality-sensitive hashing (Andoni et al., 2006) as a recent method developed within the computer science community. Neural associative memories are biologically inspired solutions for this problem.

We have chosen to apply our hierarchical representation scheme to the classic 'Steinbuch-Willshaw-Palm' model due to its amenity to analysis, high information efficiency and low computational complexity. If the threshold can be determined a priori, an associative memory of this kind is only prone to introducing add-errors. Hence, the 'no false negatives' property can be exploited and the approximated memories can be safely compressed beyond the traditional limit of $p_1 = 0.5$. This results in large performance improvements for any capacity setting (within the limits defined by Palm (1980)) of the uncompressed memory R. For input cues with arbitrary noise levels, other extensions or models described in the literature remain more apt, such as iterative retrieval techniques (Sommer & Palm, 1999) or associative memories of spiking neurons (Knoblauch & Palm, 2001).

From a computational point of view, our model has been shown useful as a means to enhance retrieval performance. The major contributing cost factor $n \cdot |\tilde{x}|_0$ in equation 4 can be substantially diminished thanks to the tree-like hierarchical lookup procedure, spending less than twice the original space. The total number of synapses is clearly limited for finite hierarchical memories, as we have $\sum_{r=1}^{R-1} mn/2^r + mn < 2mn$ for the least compressed case, i.e., when $a_1 = \ldots = a_{R-1} = 2$. Since the traditional measure of storage capacity is usually normalized by the number of available synaptic contacts (Knoblauch et al., 2010), this means that our model will exhibit at least half of the capacity of the original Steinbuch network.

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.

Synaptic activity on the mammalian brain has been related to increased energy costs which may not be sustainable by the underlying metabolic processes. How the proposed hierarchical refinement process could be implemented biologically remains an interesting open question.

Acknowledgements

The authors wish to express their gratitude towards Jan Cederquist and the anonymous reviewers for their helpful and incisive comments on early versions of this manuscript. The authors are also indebted to Alexander Pattenden for proofreading and suggesting useful corrections to the language. This work was supported by *Fundação para a Ciência e Tecnologia* (INESC-ID multiannual funding) through the PIDDAC Program funds and through an individual doctoral grant awarded to the first author (contract SFRH/BD/66398/2009).

Appendix A. Estimating the retrieval cost of our model

Following the analytical treatment due to Palm (1980) where the case of uniform random patterns (which minimize pattern superposition and maximize memory capacity) is dealt with, it is possible to estimate the computational cost t^{hier} of our hierarchical retrieval procedure.

The total retrieval cost can be defined trivially as the sum of the retrieval costs for every level r from 1 to R:

$$t^{\text{hier}} = t_1 + t_2 + \ldots + t_{r+1} + \ldots + t_R. \tag{A.1}$$

For the first memory at $r = 1$ where a full lookup is performed, equation 4 can be used to calculate the exact cost, resulting in

$$t_1 := n/(a_1 \cdot \ldots \cdot a_R) \cdot |\tilde{x}|_0 \tag{A.2}$$

operations if we do not take the threshold cuts into account. For the subsequent levels, however, we must modify the original cost function as the dendritic potentials are restricted to a subset of neurons (recall equation 11).

Note that the number of traversed neurons at level $r+1$ corresponds to the cardinality of the set $\mathcal{Y}_{r+1}$; that this set has exactly $a_{r+1} \cdot |y(r)|_0$ elements; and that the cost at that level is given by

$$t_{r+1} := a_{r+1} \cdot |y(r)|_0 \cdot |\tilde{x}|_0. \tag{A.3}$$

In other words, the remaining question is how to determine $|y(r)|_0$, i.e., the number of '1' components in the retrieved vector at level r. The answer to this problem lies in finding the dendritic potential probability distribution, a classic subject of associative memory analysis (see for instance Palm (1980); Graham & Willshaw (1995); Sommer & Palm (1999)), which has received full treatment recently in Knoblauch (2008).

The latter work derives exact formulas which calculate the neural 'firing' probability p for the most relevant scenarios. We denote p_r as the probability p at level r. As every approximated memory is necessarily hetero-associative by design thanks to the horizontal aggregation, $|y(r)|_0 = p_r \cdot n/(a_r \cdot a_{r+1} \cdot \ldots \cdot a_R)$ can be determined using either p_{Ph} or p_{Wh} from Knoblauch (2008). For our experimental case where $k := |x^\mu|_0$ and $l := |y^\mu|_0$ are fixed, we can employ directly the expression for p_{Ph}

$$\begin{aligned} &p_{\text{Ph}}(\Theta; k, l, m, n, M, z) = \\ &= \binom{z}{\Theta} \sum_{s=0}^{\Theta} (-1)^s \binom{\Theta}{s} \Big(1 - \frac{l}{n}(1 - B(m, k, s + z - \Theta))\Big)^M \end{aligned} \tag{A.4}$$

in order to derive p_r

$$p_r \approx p_{\text{Ph}}(\Theta; k, l, m, n/(a_r \cdot a_{r+1} \cdot \ldots \cdot a_R), M, z) \tag{A.5}$$

where $z := |\tilde{x}|_0$, $B(a, b, c) := \binom{a-b}{c} / \binom{a}{c}$ and where for brevity the noise term $\tilde{p}_1$ was suppressed. Equation A.5 is an approximation since we assume that l remains constant across all levels thanks to the logarithmic sparseness. This assumption can probably be refined, as the sparseness degree is actually reduced from level to level.

References

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

Andoni, A., Datar, M., Immorlica, N., Indyk, P., & Mirrokni, V. (2006). Locality-sensitive hashing scheme based on p-stable distributions. In T. Darrell, P. Indyk, & G. Shakhnarovich (Eds.), *Nearest Neighbor Methods in Learning and Vision: Theory and Practice*. MIT Press.

Bentz, H. J., Hagstroem, M., & Palm, G. (1989). Information storage and effective data retrieval in sparse matrices. *Neural Networks*, 2(4), 289–293.

Graham, B. & Willshaw, D. (1995). Improving recall from an associative memory. *Biological Cybernetics*, 72(4), 337–346.

Graham, B. & Willshaw, D. J. (1994). Capacity and information efficiency of a brain-like associative net. In G. Tesauro, D. S. Touretzky, & T. K. Leen (Eds.), *Advances in Neural Information Processing Systems 7* (pp. 513–520).

Hebb, D. O. (1949). *The organization of behavior: a neuropsychological theory*. New York: Wiley.

Hirahara, M., Oka, N., & Kindo, T. (2000). A cascade associative memory model with a hierarchical memory structure. *Neural Networks*, 13(1), 41–50.

Kakeya, H. & Kindo, T. (1996). Hierarchical concept formation in associative memory composed of neuro-window elements. *Neural Networks*, 9(7), 1095–1098.

Knoblauch, A. (2003). Optimal matrix compression yields storage capacity 1 for binary Willshaw associative memory. In O. Kaynak, E. Alpaydin, E. Oja, & L. Xu (Eds.), *Artificial Neural Networks and Neural Information Processing — ICANN/ICONIP 2003* (pp. 325–332).: Springer.

Knoblauch, A. (2008). Neural associative memory and the Willshaw–Palm probability distribution. *SIAM Journal on Applied Mathematics*, 69(1), 169–196.

Knoblauch, A. & Palm, G. (2001). Pattern separation and synchronization in spiking associative memories and visual areas. *Neural Networks*, 14(6-7), 763–780.

Knoblauch, A., Palm, G., & Sommer, F. T. (2010). Memory capacities for synaptic and structural plasticity. *Neural Computation*, 22(2), 289–341.

Laughlin, S. B. (2001). Energy as a constraint on the coding and processing of sensory information. *Current Opinion in Neurobiology*, 11(4), 475–480.

Lennie, P. (2003). The cost of cortical computation. *Current Biology*, 13(6), 493–497.

Nadal, J.-P. & Toulouse, G. (1990). Information storage in sparsely coded memory nets. *Network: Computation in Neural Systems*, 1, 61–74(14).

Palm, G. (1980). On associative memory. *Biological Cybernetics*, 36(1), 19–31.

Palm, G. & Palm, M. (1991). Parallel Associative Networks: The PAN-system and the Bacchus-Chip. In U. Ramacher, U. Rückert, & J. Nossek (Eds.), *International Conference on Microelectronics for Neural Networks* (pp. 411–416).: Kyrill & Method.

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).

Palm, G. & Sommer, F. T. (1996). Associative data storage and retrieval in neural networks. In *Models of Neural Networks: Association, Generalization, and Representation (Physics of Neural Networks)*, volume 3 (pp. 79–118). Springer, New York.

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.

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 F. Roli & S. Vitulano (Eds.), *Image Analysis and Processing — ICIAP 2005* (pp. 786–793).: Springer.

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

Wichert, A. (2009). Subspace tree. In *Seventh International Workshop on Content-Based Multimedia Indexing* (pp. 38–43).: IEEE Computer Society.

Willshaw, D. J., Buneman, O. P., & Longuet-Higgins, H. C. (1969). Non-holographic associative memory. *Nature*, 222(5197), 960–962.