INSTITUTE OF PHYSICS PUBLISHING NETWORK: COMPUTATION IN NEURAL SYSTEMS

Network: Comput. Neural Syst. **13** (2002) 179–194 PII: S0954-898X(02)35227-8

A Bayesian attractor network with incremental learning

A Sandberg[1], **A Lansner**[1], **K M Petersson**[2] **and Ö Ekeberg**[1]

[1] Department of Numerical Analysis and Computing Science, Royal Institute of Technology, 100 44 Stockholm, Sweden
[2] Cognitive Neurophysiology Research Group, Department of Clinical Neuroscience, Karolinska Institutet, Stockholm, Sweden

E-mail: asa@nada.kth.se, ala@nada.kth.se, karl.magnus.petersson@ks.se and orjan@nada.kth.se

Received 8 June 2001
Published 30 April 2002
Online at stacks.iop.org/Network/13/179

Abstract

A realtime online learning system with capacity limits needs to gradually forget old information in order to avoid catastrophic forgetting. This can be achieved by allowing new information to overwrite old, as in a so-called palimpsest memory. This paper describes an incremental learning rule based on the Bayesian confidence propagation neural network that has palimpsest properties when employed in an attractor neural network. The network does not suffer from catastrophic forgetting, has a capacity dependent on the learning time constant and exhibits faster convergence for newer patterns.

1. Introduction

In contrast to the arranged learning situation often faced by artificial neural networks (ANN), real world learning presents a next to unlimited number of learning examples, potentially surpassing the storage capacity of the learner by orders of magnitude. An additional complication is that the world may be non-stationary, partly due to the changing behaviour of the system and its interaction with the environment. In general, it is essential for a capacity limited real world learning system to give priority to the retention of recent information of a kind that is relevant to its operation. Several different timescales may be involved, as in for example short-term and long-term memory. These are fundamental constraints for existing biological learning and memory systems and are also critical for advanced artificial learning systems.

Auto-associative attractor ANNs, for example early binary associative memories and the more recent Hopfield net, have been proposed as models for biological associative memory [1–3]. They can be regarded as formalizations of Hebb's original ideas of synaptic plasticity and emerging cell assemblies [4]. A number of psychological memory and Gestalt perception phenomena have been modelled based on such networks [5, 6].

The recurrent architecture of attractor networks corresponds to the large-scale as well as the local patterns of connectivity of the cerebral cortex. There one finds a local and medium-range horizontal network as well as long-range feed-forward, feed-back and lateral connectivity between cortical areas [7,8]. Simulations have indicated that a network of cortical pyramidal and basket cells can perform well as an attractor network [9–13]. Specifically, a network using the counting Bayesian learning rule [14] discussed below implemented with biologically detailed compartmental model neurons with cortical minicolumns as the functional units operates as an attractor neural network. Moreover, it has a local connectivity that is dense with a sparse long-range connectivity and when scaling up this network model to a patch of 8×8 mm cortex one obtains cell counts and connectivity patterns (including EPSP and IPSP amplitudes) compatible with experimental data on cortical micro-architecture [15].

1.1. Palimpsest memory

Standard correlation based learning rules for attractor ANNs suffer from catastrophic forgetting, that is, all memories are lost when the system becomes overloaded. To cope with this situation Nadal *et al* [16] proposed a so-called 'marginalist' learning paradigm where the acquisition intensity is tuned to the present level of crosstalk 'noise' from other patterns. This makes the most recently learned pattern the most stable; new patterns are stored on top of older ones, which are gradually overwritten and become inaccessible, a so-called 'palimpsest memory'.

Another smoothly forgetting learning scheme is 'learning within bounds' (originally suggested by Hopfield [2]), where the connection weights w_{ij} are bounded $-A \leqslant w_{ij} \leqslant A$. The learning rule for training patterns ξ^n is

$$w_{ij}(n+1) = c\left(w_{ij}(n) + \frac{\xi_i^n \xi_j^n}{\sqrt{N}}\right)$$

where c is a clipping function

$$c(x) = \begin{cases} -A & x < -A \\ x & |x| \leqslant A \\ A & A < x. \end{cases} \tag{1}$$

For high values of A catastrophic forgetting occurs, for low values the network remembers only the last pattern. The optimal capacity $0.05N$ is reached for $A \approx 0.4$ [17,18]. This implies a decrease in storage capacity from $0.137N$ of the standard Hebbian rule; total capacity has been sacrificed for long-term stability.

1.2. Bayesian confidence propagation neural networks (BCPNN)

A neural network architecture and learning rule derived from Bayes' rule [19], the Bayesian confidence propagation neural network (BCPNN), has previously been developed [14,20–23]. It employs a Hebbian learning rule that reinforces connections between simultaneously active units and weakens or makes connections inhibitory between anti-correlated units. This learning rule is based on a probabilistic view of learning and retrieval, with input and output unit activities representing confidence of feature detection and posterior probabilities of outcomes, respectively. The connection strengths are based on the probabilities of the units firing together, estimated by counting co-occurrences in the training data. When applied to a recurrent attractor network this learning rule gives a symmetric weight matrix, allowing for fixed point attractor dynamics. It also generates a proper balance between excitation and inhibition, avoiding the

need for external means of threshold regulation. The update of the weights in the network resembles what has been proposed as rules for biological synaptic plasticity [24, 25].

It should be noted that the type of Bayesian neural network studied here differs from, for example, those proposed by MacKay [26] and Sommer and Dayan [27]. It is more closely related to methods of Bayesian abduction used in AI [28] and the Bayesian neural network model proposed by Kononenko [20].

In this paper, we demonstrate that by estimating the probabilities underlying network biases and weights by moving averages instead of counters as in the previous versions, it is possible to derive a continuous, real-time Bayesian learning rule with the properties of a palimpsest memory. The forgetfulness of the network can conveniently be regulated by the time constant of the moving averages. We evaluate this learning rule in the context of long-term and short-term memory properties of an attractor network with some biologically plausible elements. The consequences of a time-dependent energy landscape in terms of convergence speed and plasticity modulation are also investigated.

2. Heuristic derivation of network architecture and learning rule

The BCPNN is based on an update rule heuristically derived from Bayes rule and the naive Bayesian classifier (NBC) [29].

2.1. Naive Bayesian classifier BCPNN

We start with the NBC, calculating the probabilities of the attributes y_j given a set $\boldsymbol{x}$ of observed attributes x_i. Both are assumed to be discrete, and x_i are assumed to be independent ($P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i)$) and conditionally independent given y_j ($P(x_1, \ldots, x_n|y_j) = \prod_{i=1}^{n} P(x_i|y_j)$).

Using Bayes' rule we get

$$\pi_j = P(y_j|\boldsymbol{x}) = P(y_j)\prod_{i=1}^{n}\frac{P(x_i|y_j)}{P(x_i)} = P(y_j)\prod_{i=1}^{n}\frac{P(x_i, y_j)}{P(y_j)P(x_i)}.$$

This can be extended to the case where some information is not known. Suppose we are given completely known observations x_i when $i \in A \subseteq \{1, \ldots, n\}$ and have no information about the attributes x_k when $k \in \{1, \ldots, n\} - A$, then for a NBC

$$\pi_j = P(y_j|x_i, i \in A) = P(y_j)\prod_{i\in A}\frac{P(y_j, x_i)}{P(y_j)P(x_i)}. \tag{2}$$

This can be put in the form of a sum of logarithms

$$\log \pi_j = \log P(y_j) + \sum_{i\in A}\log\left[\frac{P(y_j, x_i)}{P(y_j)P(x_i)}\right] = \log P(y_j) + \sum_{i}^{n} o_i \log\left[\frac{P(y_j, x_i)}{P(y_j)P(x_i)}\right]$$

where the indicator variable

$$o_i = \begin{cases} 1 & i \in A \\ 0 & i \notin A \end{cases} = I_A(i)$$

represents whether there is information about a feature i or not.

This can be implemented as a single-layer feedforward neural network, with input layer activations o_i, weights $w_{ij} = \log\left[\frac{P(y_j,x_i)}{P(y_j)P(x_i)}\right]$ and biases $\beta_j = \log P(y_j)$. In this way the single-layer feedforward neural network calculates posterior probabilities π_j given the input attributes using an exponential transfer function.

2.2. Discrete valued attribute network

If discrete attribute values are represented using indicator variables, a modular structure of the BCPNN follows. Continuous valued attributes can be interval coded, a coding principle abundant in the nervous system. One classical example is the coding of edge orientation in the primary visual cortex [30], where each orientation minicolumn responds selectively to an interval of orientations. The orientation hypercolumn contains orientation columns covering all angles, and thus represents the local edge orientation pertinent to a given point in visual space. A similar modular arrangement is found in blobs and many other cortical areas, e.g. rodent whisker barrels [31]. By analogy with this possibly generic cortical structure we have referred to our network model as having a hypercolumnar structure. It should be noted that one BCPNN unit maps naturally to a minicolumn rather than to an individual neuron.

Suppose that each attribute i can take M_i different values, and that we treat the observation of a given value of a given attribute as a new binary attribute marked with double indices, the first indicating the attribute and the second the particular value. Making the necessary labellings in formula 2 we get

$$\pi_{jj'} = P(y_{jj'}) \prod_A \frac{P(y_{jj'}, x_{ik})}{P(y_{jj'})P(x_{ik})}$$

where for each attribute $i \in \{1, \ldots, n\}$ a unique value x_{ik} is known, where $k \in \{1, \ldots, M_i\}$. Similarly it follows that

$$\pi_{jj'} = P(y_{jj'}) \prod_{i=1}^{n} \sum_{i'=1}^{M_i} \frac{P(y_{jj'}, x_{ii'})}{P(y_{jj'})P(x_{ii'})} o_{ii'}$$

with indicators $o_{ii'} = 1$ if $i' = k$ and zero otherwise.

If we consider the attributes X_i as stochastic variables with values $\{x_{i1}, \ldots, x_{iM_i}\}$, which are explicitly represented in the network, we may view $o_{X_i}(x_{ii'}) := o_{ii'}$ as a degenerate probability $o_{X_i}(x_{ii'}) = \delta_{x_{ik}}(x_{ii'})$ which is zero for all $x_{ii'}$ except for the known value x_{ik}. If we now generalize and replace o_{X_i} with a general probability P_{X_i} we get

$$\hat{\pi}_{jj'} = P(y_{jj'}) \prod_{i=1}^{n} \sum_{i'=1}^{M_i} \frac{P(y_{jj'}, x_{ii'})}{P(y_{jj'})P(x_{ii'})} P_{X_i}(x_{ii'})$$

$$\log(\hat{\pi}_{jj'}) = \log P(y_{jj'}) + \sum_{i=1}^{n} \log \left[\sum_{i'=1}^{M_i} \frac{P(y_{jj'}, x_{ii'})}{P(y_{jj'})P(x_{ii'})} P_{X_i}(x_{ii'}) \right].$$

If the outcomes $x_{ii'}$ of different attributes are independent of each other when conditioned on X_i, $\hat{\pi}_{jj'}$ will be the expectation of $\pi_{jj'}$ given the input X_i.

The corresponding network now has a modular structure. The units ii' in the network, where $i' \in \{1, \ldots, M_i\}$, explicitly representing the values $x_{ii'}$ of X_i may be viewed as a *hypercolumn* as discussed above. By definition the units of a hypercolumn i have a normalized total activity $\sum_{i'=1}^{M_i} P_{X_i}(x_{ii'}) = 1$.

These procedures estimate the probability of $y_{jj'}$ given uncertain information related to $x_{ii'}$. In this case the uncertainty of the attribute is reflected in the probability $P_{X_i}(x_{ii'})$ which is the input to the network.

Transforming these equations to the network setting yields

$$h_{jj'} = \beta_{jj'} + \sum_{i}^{N} \log \left(\sum_{i'}^{M_i} w_{ii'jj'} P_{X_i}(x_{ii'}) \right) \tag{3}$$

where $h_{jj'}$ is the *support* of unit jj'. We make the identifications

$$\beta_{jj'} = \log(P(y_{jj'})) \tag{4}$$

$$w_{ii'jj'} = \frac{P(x_{ii'}, y_{jj'})}{P(x_{ii'})P(y_{jj'})} \tag{5}$$

where $\beta_{jj'}$ is the bias term and $w_{ii'jj'}$ is the weight. $\hat{\pi}_{jj'} = f(h_{jj'}) = \mathrm{e}^{h_{jj'}}$ can be identified as the output of unit jj', representing the *confidence* (heuristic or approximate probability) that attribute j has value j' given the current context.

Since the independence assumption is often only approximately fulfilled and we deal with approximations of probabilities, it is motivated to normalize the output within each hypercolumn:

$$\hat{\pi}_{jj'} = f(h_{jj'}) = \frac{\mathrm{e}^{h_{jj'}}}{\sum_{j'} \mathrm{e}^{h_{jj'}}}. \tag{6}$$

2.3. Recurrent network

Now, since both the input and the output of the network represent probabilities this opens up the possibility of feeding the output back into the network as input creating a fully recurrent network architecture, which can work as an autoassociative memory. The currently observed probability $P_{X_i}(x_{ii'})$ is used as an initial approximation of the true probability of $X_{ii'}$, and used to calculate a posterior probability, which tends to be a better approximation. This is then fed back and the process is iterated until a consistent state is reached. This represents a heuristically estimated probability closest to the observed data and consistent with the already acquired knowledge, that is prior information represented by the learning parameters, $\beta_{jj'}$ and $w_{ii'jj'}$. The weight matrix is symmetric and the iteration converges towards an attractor state.

In the recurrent setting activations in the network can be updated either discretely or continuously (observe that the $y_{jj'}$ are now incorporated among the $x_{ii'}$). In the discrete case, $\hat{\pi}_{jj'}(t+1)$ is calculated from $\hat{\pi}_{ii'}(t)$, or equivalently, the $h_{jj'}(t+1)$ from $h_{ii'}(t)$ using one iteration of the update rule

$$h_{jj'}(t+1) = \beta_{jj'} + \sum_{i}^{N} \log\left(\sum_{i'}^{M_i} w_{ii'jj'} f(h_{ii'}(t))\right).$$

In the continuous case $h_{jj'}(t)$ is updated according to a differential equation, making the approach towards an attractor state continuous:

$$\tau_c \frac{\mathrm{d}h_{jj'}(t)}{\mathrm{d}t} = \beta_{jj'} + \sum_{i}^{N} \log\left(\sum_{i'}^{M_i} w_{ii'jj'} f(h_{ii'}(t))\right) - h_{jj'}(t) \tag{7}$$

where τ_c is the 'membrane time constant' of each unit.

The network is used with a training mode where the weights are set and a retrieval mode where inferences are made. Input to the network is introduced by clamping the activation of the relevant units (representing known events or attributes). As the network is updated the activation spreads, creating *a posteriori* beliefs of other attribute values.

2.4. Incremental Bayesian learning

The original formulation of the BCPNN learning rule is based on estimating the probabilities $P(x_{ii'})$ from a given training set using counters [14, 22]. The number of instances when each particular unit is active in the training set are counted, and the counter divided by the

total number of training examples is used as an approximation of the probability $P(x_{ii'})$. Co-activations of two units are counted similarly, estimating $P(x_{ii'}, x_{jj'})$. Once these estimates have been calculated, the weights and biases are calculated and the network can be used in retrieval mode.

A continuously operating network will need to learn incrementally during operation. In order to achieve this, $P(x_{ii'})(t)$ and $P(x_{ii'}, x_{jj'})(t)$ need to be estimated given the information $\{\boldsymbol{x}(t'), t' < t\}$. What we aim for is an estimate with the following properties:

(i) It should converge towards $P(x_{ii'})(t)$ and $P(x_{ii'}, x_{jj'})(t)$ in a stationary environment.
(ii) It should give more weight to recent than remote information.
(iii) It should smooth or filter out noise and adapt to longer trends, in other words lower frequency components of a non-stationary environment.

The incremental Bayesian learning rule proposed here achieves this by approximating $P(x_{ii'})(t)$ and $P(x_{ii'}, x_{jj'})(t)$ with the exponentially smoothed running averages $\Lambda_{ii'}$ of the activity $\hat{\pi}_{ii'}$ and $\Lambda_{ii'jj'}$ of coincident activity $\hat{\pi}_{ii'}\hat{\pi}_{jj'}$.

The continuous time version of the update and learning rule takes the following form:

$$\tau_c \frac{\mathrm{d}h_{ii'}(t)}{\mathrm{d}t} = \beta_{ii'}(t) + \sum_j^N \log\left(\sum_{j'}^{M_i} w_{ii'jj'}(t)\hat{\pi}_{jj'}(t)\right) - h_{ii'}(t) \tag{8}$$

$$\hat{\pi}_{ii'}(t) = \frac{\mathrm{e}^{h_{ii'}}}{\sum_j \mathrm{e}^{h_{ij}}} \tag{9}$$

$$\frac{\mathrm{d}\Lambda_{ii'}(t)}{\mathrm{d}t} = \alpha([(1-\lambda_0)\hat{\pi}_{ii'}(t) + \lambda_0] - \Lambda_{ii'}(t)) \tag{10}$$

$$\frac{\mathrm{d}\Lambda_{ii'jj'}(t)}{\mathrm{d}t} = \alpha([(1-\lambda_0^2)\hat{\pi}_{ii'}(t)\hat{\pi}_{jj'}(t) + \lambda_0^2] - \Lambda_{ii'jj'}(t)) \tag{11}$$

$$\beta_{ii'}(t) = \log(\Lambda_{ii'}(t)) \tag{12}$$

$$w_{ii'jj'}(t) = \frac{\Lambda_{ii'jj'}(t)}{\Lambda_{ii'}(t)\Lambda_{jj'}(t)}. \tag{13}$$

The learning rate $\alpha = 1/\tau_L$ is the inverse of the learning time constant; it is a more convenient parameter than τ_L. By setting α temporarily to zero the network activity can change with no corresponding weight changes, for example during the retrieval mode.

To avoid logarithms of zero in the calculations, a basic low-activity $\lambda_0 \ll 1$ is introduced, a kind of noisy background activity that is present regardless of external signals. In the absence of input $\Lambda_{ii'}(t)$ and $\Lambda_{jj'}(t)$ now converge towards λ_0 and $\Lambda_{ii'jj'}(t)$ towards λ_0^2, producing $w_{ii'jj'}(t) = 1$ for large t (corresponding to uncoupled units). The smallest possible weight value if the state variables are initialized to λ_0 and λ_0^2 respectively, is $4\lambda_0^2$, and the smallest possible bias $\log(\lambda_0)$. The upper bound on the weights becomes $1/\lambda_0$. In the following, $\lambda_0 = 0.0001$.

The above probability estimates converge towards the correct values given stationary inputs for sufficiently large time constants. Since the weights of the network depend more on recent than on old data, it appears likely that a Hopfield-like network with the above learning rule would exhibit palimpsest properties.

3. Results

3.1. Behaviour of single-connection weights

We first study how the weight $w_{ii'jj'}$ between units ii' and jj' changes with correlation of unit activities. A discretized version of the learning rule was used.

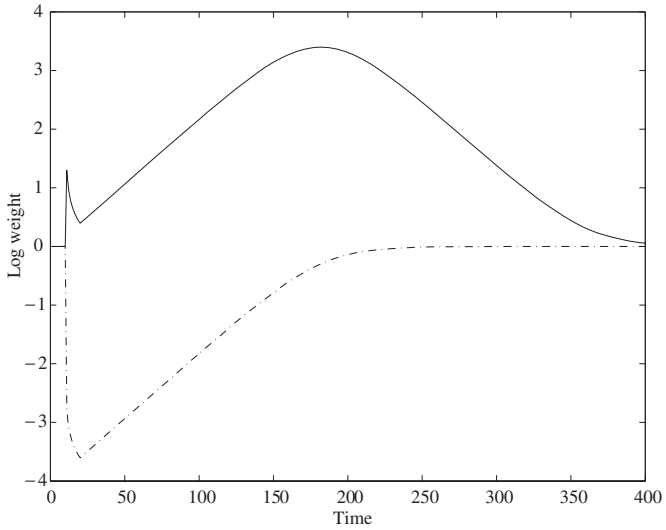


Figure 1. The weight between two units that are active together at $10 \leqslant t \leqslant 20$ (solid curve) and when only one unit is active (dot–dash curve) for $\alpha = 0.05$. Note the logarithmic y-scale.

The behaviour of $w_{ii'jj'}$ can be seen in figure 1 for two correlated and two anti-correlated units. When both units are active together for a period of time the connection is strengthened significantly, but the strength of the connection decreases during prolonged stimulation. After the stimulation the weight begins to increase again, a result of the fact that the product $\Lambda_{ii'}\Lambda_{jj'}$ decays faster than $\Lambda_{ii'jj'}$. This continues until the estimates level out and the weight goes to zero. It should be noted that the increase in weights is balanced by the behaviour of the bias; the network dynamics remains stable despite the strong weight changes.

3.1.1. Non-stationary activities. The above situation of two units firing together against a background of very low activity is somewhat extreme given the assumptions of the model. A more canonical example of the changes in weights due to randomly firing units as well as the response to non-stationary activities can be seen in figure 2 where the two units are correlated, uncorrelated or anti-correlated with each other at different times. As can be seen, after a brief transient the weight moves towards a steady state value of 2 as the units remain correlated. As the correlation vanishes the log weight begins to move randomly with a mean close to zero. When the units become anti-correlated the logarithm of the weight decreases practically linearly towards the baseline negative value of $4\lambda_0^2 = 4 \times 10^{-8}$. Finally, when the correlations reappear, the weight quickly increases to the steady state value.

When a network is trained with a set of random patterns the logarithms of the weights between two units may change sign depending on whether the units are activated by different patterns (negative logarithm of the weight) or the same pattern (positive log weight) as can be seen in figure 3.

3.2. Network behaviour

In the following experiments a network with 100 units (organized as ten hypercolumns) was first trained by the repeated presentation of the training patterns. Unit activities were clamped to the input patterns for one unit of time. This was followed by a testing period where α was

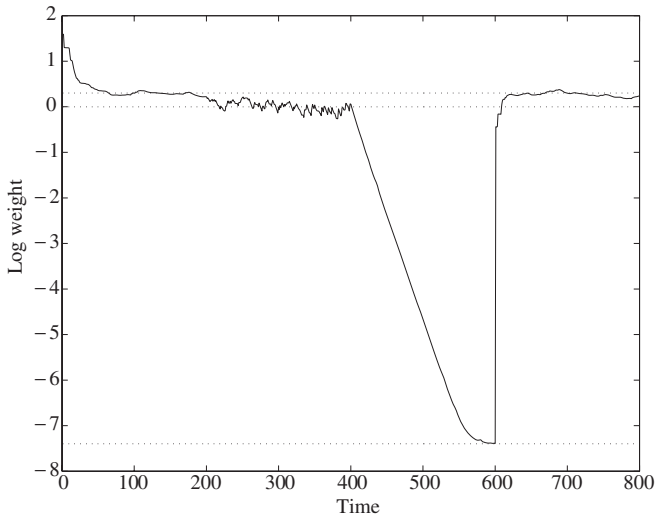


Figure 2. Weight between two units that are active 50% of the time, completely correlated for $0 \leqslant t \leqslant 200$, uncorrelated for $200 \leqslant t \leqslant 400$, completely anti-correlated for $400 \leqslant t < 600$ and finally correlated again. The dotted lines correspond to the predicted values $\log(2)$, 0 and $\log(4\lambda_0^2)$. In this run $\alpha = 0.05$.

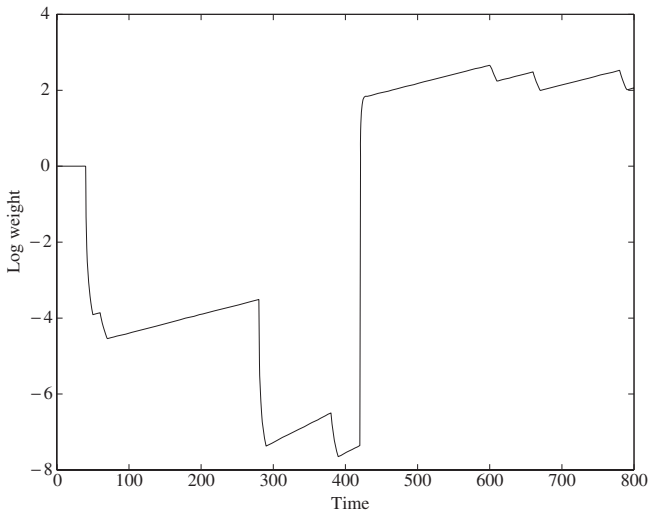
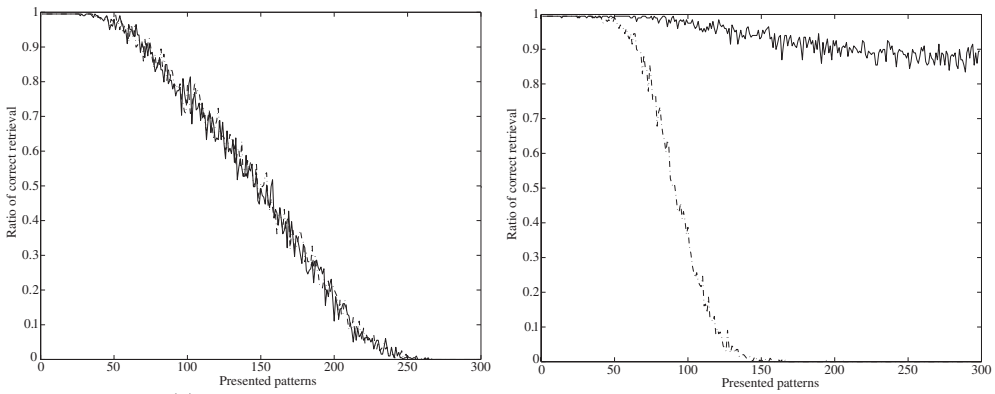


Figure 3. Weight between two units during the training of the network with sparse random patterns (10% activity, ten timesteps of presentation of each pattern followed by ten steps of decay with no activity) with $\alpha = 0.05$. At time 40 one of the units is recruited into a pattern while the other remains silent, causing a strong inhibitory connection to develop. At time 420 both units become recruited by the same pattern, making the connection positive. It then remains positive for the rest of the run, despite occasional activations and coactivations.

set to zero and no learning took place. The continuous update rule from equations (8)–(13) was solved using Euler's method with step length $h = 0.1$, $\tau_c = 1$.

The performance for a given pattern $\boldsymbol{\xi}$ was measured as the percentage of perturbed patterns (the activations of two previously active units have randomly been swapped with the activation



(*a*) Non-incremental (*b*) Incremental

Figure 4. Comparison of the previous non-incremental BCPNN learning rule and the incremental learning rule for sparse random patterns ($\rho = 0.1$, one active unit per hypercolumn). The ratio of correctly retrieved patterns (overlap > 0.85) is shown for the first learned pattern $\boldsymbol{\xi}_1$ and the latest $\boldsymbol{\xi}_n$. Solid curve: latest learned pattern, dash-dotted curve: first learned pattern. $\alpha = 0.01$ in the incremental case.

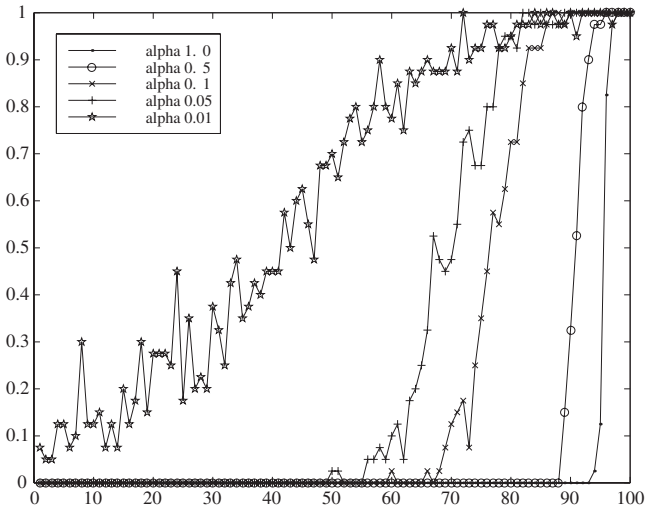


Figure 5. Forgetting curve after incremental learning. Pattern recall as a function of list position for different values of α (see legend) in a network with 100 units. Recall estimated as the frequency of retrieval with overlap greater than 0.85 after a presentation of a pattern where two activation of two hypercolumns had been randomly changed followed by 1.0 time units of convergence. Random patterns, 10% activation, 1.0 time units of presentation.

of other units) which were correctly recalled after relaxation to a tolerance of 0.85 overlap (the overlap was defined to be $\boldsymbol{\xi} \cdot \hat{\pi} / ||\boldsymbol{\xi}|| ||\hat{\pi}||$).

Figure 4 shows a comparison between a counter and incremental version of BCPNN. As can be seen the incremental learning rule avoids catastrophic forgetting by forgetting the oldest patterns while the recent patterns remain accessible. Figure 5 shows this in more detail. The forgetting does not occur immediately: for longer learning time constants the pattern is stored well until a certain number of interfering patterns have been stored, when it starts to gradually fade.

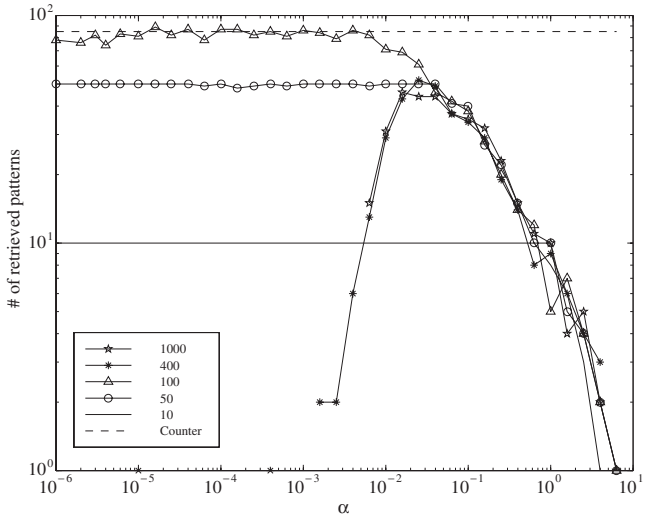


Figure 6. The number of correctly retrieved patterns as a function of α for different sizes of the training set. 10, 50, 100, 200, 400 and 1000 patterns were used in the training set, which was repeated 20 times. Successful retrieval is defined as in figure 5. As a comparison the maximal capacity of the counter model trained with 100 patterns is drawn as a dashed line.

Figure 6 shows the number of retrievable patterns as a function of α and the size of the training set. For large α the number of retrieved patterns is independent of the size of the training set. For small α a number of patterns up to the maximum capacity of the counter model can be retrieved. For much larger training sets catastrophic forgetting occurs (as for 400 and 1000 patterns when $\alpha < 10^{-3}$): none of the patterns can be retrieved due to mutual interference.

The same situation but with only one presentation of the training set instead of repetition gives a slightly different picture. Here, even for small training set sizes, the number of patterns retrieved drops for α of about 10^{-4} and below. The reason is that the learning becomes so slow that repeated training is necessary for encoding. For α larger than 10^{-2} the behaviour is identical with or without repetition, i.e. the most recently presented patterns are remembered well, regardless of training set size. For large training sets the capacity reaches about 60% of the maximum capacity. As can be seen in figure 6, for the network simulated this occurs for values of α approximately 0.02. In summary, for small α the network operates as a long-term memory of a traditional form, such as the counter BCPNN, thus suffering from catastrophic forgetting. For large α it works as a short-term memory with a storage capacity limited by forgetting in the form of weight decay. By modulating α we can tune system performance continuously between these two extremes.

3.3. Convergence speed

Besides capacity and pattern completion, another characterizing property of an attractor neural network is the convergence properties and structure of its state-space. Especially when viewed as a model for working memory retrieval speed becomes relevant. Figures 7 and 8 show convergence times for the network as a function of age of patterns and as a function of the number of patterns stored. The stopping criterion used was that the rate of change was below a certain level $\|\mathrm{d}\hat{\boldsymbol{\pi}}/\mathrm{d}t\|_1 < 0.05$. Trials where convergence did not occur within 3 time units or where it converged to the wrong attractor were not counted.

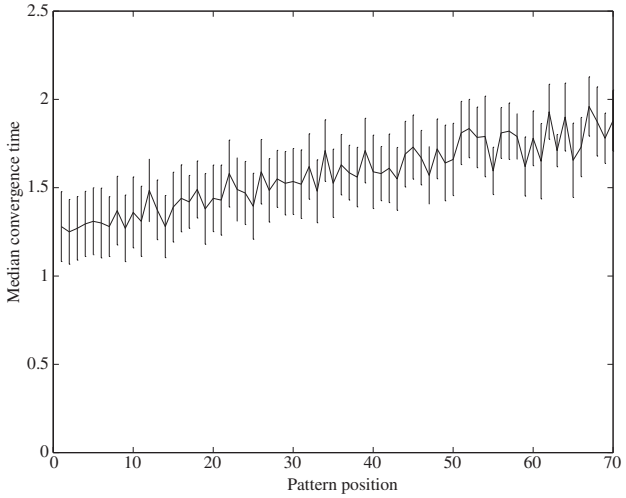


Figure 7. Median convergence time and standard deviation for retrieval of stored patterns from a disturbed input, as a function of position in the training set. Pattern 1 is the most recently learned pattern. Only retrieval of correct patterns was counted, and since this was sparser beyond pattern 70 only the first 70 patterns are shown. $\alpha = 0.01$, the network was trained with 100 patterns.

Using this learning rule, as more patterns are learned, the basins of attraction of old patterns become smaller and 'shallower' in terms of the energy landscape, eventually disappearing altogether. This also results in a change in convergence speed. There is both a difference between patterns, the latest patterns are completed faster than older patterns, and a roughly linear increase in the median convergence time between networks where few patterns have been learned and networks where the capacity has been reached (figures 7 and 8). Once the network has reached its maximal capacity for a given τ_L, the convergence time remains roughly constant.

Figure 9 shows convergence times when the network is started with a mixture between two patterns (the mixing consists of using 0–10 hypercolumns from one of them and the rest from the other pattern). For the two most recently learned patterns the convergence time is maximal at a 50% mixture (this is practically identical to the result in [14] for the non-incremental Bayesian learning rule). When interpolated between a recent and an old memory the maximum is moved away from the recent memory, a sign that the old memory has a smaller and weaker attractor than the newer memory. It can, however, still be retrieved given a similar enough cue.

3.4. Free recall

When stimulated with noise (randomly activated units with the same density as the patterns) the network either converges to one of the learned patterns or to a spurious attractor state. This could be viewed as an abstract model of free recall in memory tests. The probability of convergence towards a certain memory decreases with its age (figure 10(*a*)). This is in accordance with the results on learning within bounds due to Geszti and Pázmándi [32]. They found that as more patterns were stored the basins of attraction of old patterns were more and more flattened energy-wise, and relaxation tended to move the system to a deeper attractor (i.e. a recent pattern).

By modulating α during learning of individual patterns recall can be selectively enhanced or inhibited. Figure 10(*b*) shows the enhancing effect of a selective increase in α for a single pattern on recall. Depending on the baseline α and the relative increase various forgetfulness effects such as proactive and retroactive inhibition of other patterns can be demonstrated, similar to observed memory effects in the von Restorff learning paradigm [33].

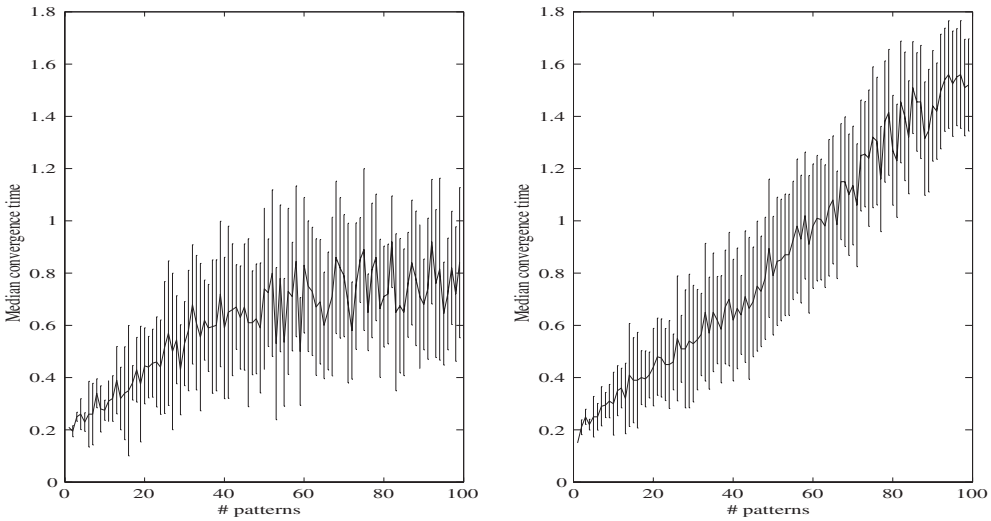


Figure 8. Median convergence time and standard deviation for increasing memory load for $\alpha = 0.1$ (left) and $\alpha = 0.01$ (right). The network is trained with up to 100 patterns, but for $\alpha = 0.1$ has capacity for only around 20–30.

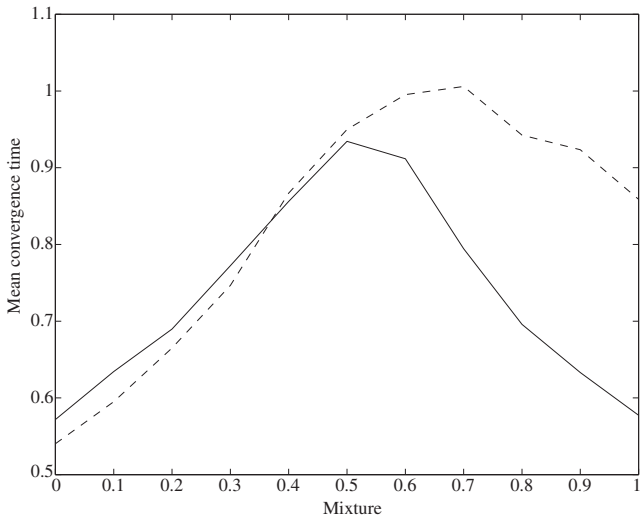


Figure 9. Convergence time for mixed input patterns. The input to the network consists of 0–10 hypercolumns of one pattern and the rest from the other pattern. The solid curve represents interpolation between the latest and second latest pattern, the dashed curve interpolation between the latest and an old pattern (#10). 70% of pattern 10 was required to produce convergence to that pattern. The network was trained with $\alpha = 0.01$.

3.5. Comparison with clipped weights

We compared our results with the performance of a Hopfield network with clipped weights similar to [17]. In order to make relevant comparisons we added sparse activation and a winner-take-all rule to the model with clipped weights.

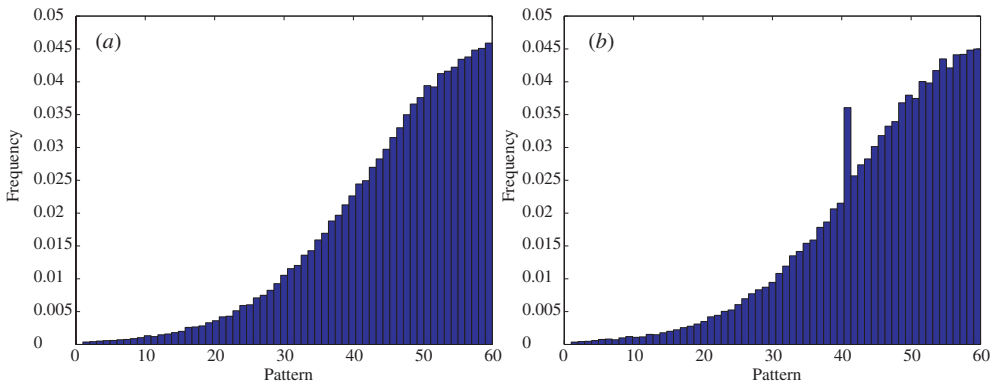


Figure 10. Frequency of ending up in the different patterns when activating the network with a random pattern (one randomly selected unit active in each hypercolumn) and allowing it to converge. In (*b*) α was temporarily increased to 0.075 for pattern 40, producing a local enhancement. Each bar corresponds to one pattern. The closest pattern to the resulting end state was used for the diagram. Only patterns with overlap greater than 0.9 are shown; they make up 0.38 of the trials. 7395 trials where the network was trained with 60 patterns and allowed to converge for 100 random patterns; $\alpha = 0.05$.

(This figure is in colour only in the electronic version)

The weights were set based on the sparse Hopfield network [34]:

$$w_{ii'jj'}(n+1) = c(w_{ii'jj'}(n) + (\xi^n_{ii'} - \rho)(\xi^n_{jj'} - \rho))$$

where c is the same clipping function as in equation (1) above and ρ is the average activity level. The update rule used was

$$\tau_c h'_{jj'}(n+1) = \sum_{ii'} w_{ii'jj'} x_{ii'}(n) - h_{jj'}(n)$$

$$x_{ii'}(n+1) = \begin{cases} 0 & h_{ii'}(n+1) < \max_k(h_{ik}(n+1)) \\ 1 & h_{ii'}(n+1) = \max_k(h_{ik}(n+1)). \end{cases}$$

This guarantees a single active unit in each hypercolumn (in cases of ties, one unit is randomly selected).

The forgetting curves (not shown) were similar in character to those presented by Parisi [17] and also those in figure 5, with the clipping range matched to the learning time constant τ_L. The number of correctly retrieved patterns from a disturbed input similar to the one used in the capacity experiments showed a similar behaviour as the incremental BCPNN. The maximal capacity in this model was around 20–30 patterns for 100 units and 400 patterns in the training set. This is significantly lower than the incremental BCPNN ($\approx$50 patterns in this case) but still significantly higher than the numbers given for the Parisi model [17], which used 50% activity and had no hypercolumns. Our conclusion is that the two models are similar in character but it appears that the BCPNN has a better performance in terms of storage capacity.

4. Discussion

We have proposed and characterized an incremental version of a previously described Bayesian learning rule [14]. The new rule allows for continuous real-time learning from a sequence of examples without leading to catastrophic forgetting. Instead, old information is gradually forgotten and only the most recent examples are retained, as in a palimpsest memory. As

expected the time for the memory decay scales roughly as the learning time constant τ_L. The memory capacity increases linearly with τ_L up to a limit where it becomes equal to the standard counter BCPNN. This means that the introduction of palimpsest properties has not reduced the maximal capacity as such. By setting the size of the network and the learning time constant the memory capacity can be regulated from a fast learning and forgetting 'working memory' to a slowly learning and forgetting 'long-term memory'. The time of convergence to a stored memory state depends both on the age of the memory and the load on the network.

The original BCPNN learning rule is interesting in that it is based on probabilities and statistics rather than being a standard Hebbian outer product rule. Furthermore, palimpsest memories like learning within bounds [2, 17] ignore new information supporting an already saturated connection, while non-supporting information will affect it; throwing out old knowledge is favoured regardless of how much positive evidence has accumulated. The learning rule proposed here does not suffer from this cut-off non-linearity and appears to have a better storage capacity. The learning rule is in some sense similar to marginalist learning [16], where new patterns are exponentially amplified and thus old patterns decay correspondingly relative to these.

Biological associative synaptic plasticity is generally assumed to be Hebbian and correlation based. This is also the case for the Bayesian–Hebbian learning rule used here. It thus falls in the same category as correlation based learning [35] and the BCM rule [36]. It exhibits a graded behaviour with multiple synapse activations as well as a more step-wise behaviour for single-synapse activation similar to experimental observations in LTP [37]. Like the above learning rules the BCPNN rule displays LTP as well as LTD, and with some modifications it provides a phenomenological model for synaptic long-term plasticity [25]. The rule presented here contains a bias term which adapts the excitability of the postsynaptic neuron alone. This relates to the phenomenon of EPSP-spike potentiation which has recently received increasing attention [38, 39].

It is interesting that this formally derived model produces a synaptic learning rule with many similarities to biologically observed phenomena. From a biological point of view it is quite reasonable to implement the learning rule with running averages as we have done here. There are several possibilities how a synapse could realize such computations within the biochemical networks and protein synthesis dependent processes involved in synaptic plasticity [40, 41].

In our simulations with a fast learning and forgetting 'working memory' we have found that the average convergence time increases significantly with memory load. This is similar to the classical finding by Sternberg of a reaction time dependence on list length [42] which has been used to support hypotheses of scanning processes underlying working memory [43]. Our results imply that the psychological phenomena can be described by an attractor network with fast learning dynamics.

It is interesting to consider the possibility of having a memory system comprised of multiple attractor networks with different learning dynamics and degrees of plasticity, as suggested by Little and Shaw [44]. A quickly adapting network would learn and remember presented objects in working memory, while a more slowly forgetting network might learn from single presentations (as in episodic long-term memory) and even slower learning and forgetting networks would average individual presentation events into a 'prototypic' semantic memory (cf [45] and [46] for two implementations). Such a memory structure is compatible with what is thought to exist in human memory systems [47].

Furthermore, an important aspect of memory is that of relevance information and print-now mechanisms. The learning rule proposed here will result in memory traces that are volatile in the absence of input since the weights are continuously changing to obey the current rate estimates. This leads to a gradual decay of memory over time even when little new information

arrives. An alternative possibility is to control the learning rate by some form of relevance or 'print-now' signal. In this case, simultaneous pre- and postsynaptic activation is not enough to result in weight changes. It is only when plasticity is enabled by the print-now signal that changes occur, equally for imprinting and decaying. With regard to our learning rule, we have found that this can easily be implemented as a change in the learning time constant (α is increased with the relevance of the situation). Such a mechanism may be closely related to neuromodulation in the brain, for example the effect of dopamine [48] and acetylcholine on synaptic plasticity [49,50]. Further experiments with a plasticity print-now signal in this model have demonstrated proactive and retroactive inhibition of other items, similar to psychological data on the von Restorff effect in list learning, where a salient item is recalled more easily at the expense of other items [33].

Taken together, the above makes a good case for pursuing further our investigation of the BCPNN model, its computational properties as well as its relations to biological and psychological processes and phenomena underlying perception, learning and memory.

Acknowledgments

Thanks to Mikael Djurfeldt for much helpful criticism. This work was supported by TFR grant no 221-97-274 and MFR grant no 12716.

References

[1] Willshaw D J, Buneman O P and Longuet-Higgins H C 1969 Non-holographic associative memory *Nature* **222** 960
[2] Hopfield J J 1982 Neural networks and physical systems with emergent collective computational abilities *Proc. Natl Acad. Sci. USA* **79** 2554–8
[3] Amit D J 1989 *Modeling Brain Function: The World of Attractor Neural Networks* (Cambridge: Cambridge University Press)
[4] Hebb D O 1949 *The Organization of Behavior* (New York: Wiley)
[5] Freeman W J 1991 The physiology of perception *Sci. Am.* **264** 78–85
[6] Quinlan P 1991 *Connectionism and Psychology. A Psychological Perspective on Connectionist Research* (New York: Harvester)
[7] Gilbert C D and Wiesel T N 1989 Columnar specificity of intrinsic horizontal and corticocortical connections in cat visual cortex *J. Neurosci.* **9** 2432–42
[8] Shepherd G M (ed) 1998 *The Synaptic Organization of the Brain* 4th edn (New York: Oxford University press)
[9] Hasselmo M E, Anderson B P and Bower J M 1992 Cholinergic modulation of cortical associative memory function *J. Neurophysiol.* **67** 1230–46
[10] Haberly L B and Bower J M 1989 Olfactory cortex: model circuit for study of associative memory? *Trends Neurosci.* **12** 258–64
[11] Amit D J and Brunel N 1995 Learning internal representations in an attractor neural network *Network* **6** 359
[12] Lansner A and Fransén E 1992 Modeling Hebbian cell assemblies comprised of cortical neurons *Network* **3** 105–19
[13] Fransén E and Lansner A 1995 Low spiking rates in a population of mutually exciting pyramidal cells *Network* **6** 271–88
[14] Lansner A and Ekeberg Ö 1989 A one-layer feedback artificial neural network with a bayesian learning rule *Int. J. Neural Syst.* **1** 77–87
[15] Fransén E and Lansner A 1998 A model of cortical associative memory based on a horizontal network of conneted columns *Network* **9** 235–64
[16] Nadal J P, Toulouse G, Changeux J P and Dehaene S 1986 Networks of formal neurons and memory palimpsests *Europhys. Lett.* **1** 535–42
[17] Parisi G 1986 A memory which forgets *J. Phys. A: Math. Gen.* **19** L617–20
[18] Bonnaz D 1997 Storage capacity of generalized palimpsests *J. Physique I* **7** 1709–21
[19] Bayes T 1958 An essay towards solving a problem in the doctrine of chances *Biometrica* **45** 296–315 (reprint of original article in *Phil. Trans. R. Soc.* **53** 370–418 1763)
[20] Kononenko I 1989 Bayesian neural networks *Biol. Cybern.* **61** 361–70

[21] Lansner A and Ekeberg Ö 1987 An associative network solving the '4-Bit ADDER problem' *IEEE 1st Int. Conf. on Neural Networks (San Diego, CA, June 1987)* ed M Caudill and C Butler pp II–549
[22] Lansner A and Holst A 1996 A higher order Bayesian neural network with spiking units *Int. J. Neural Syst.* **7** 115–28
[23] Holst A 1997 The use of a Bayesian neural network model for classification tasks *PhD Thesis* Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockholm, Sweden, *TRITA-NA-P9708*
[24] Levy W B and Desmond N L 1985 The rules of elemental synaptic plasticity *Synaptic Modification, Neuron Selectivity and Nervous System Organization* ed W B Levy, J A Anderson and S Lehmkule (Hillsdale: Lawrence Erlbaum Associates)
[25] Wahlgren N and Lansner A 2001 Biological evaluation of a Hebbian–Bayesian learning rule *Neurocomputing* **38–40** 433–8
[26] MacKay D J C 1995 Probable networks and plausible predictions—a review of practical Bayesian methods for supervised neural networks *Network* **6** 469–505
[27] Sommer F T and Dayan P 1998 Bayesian retrieval in associative memories with storage errors *IEEE Trans. Neural Netw.* **9** 705–11
[28] Charniak E and McDermott D 1985 *Introduction to Artificial Intelligence* (Reading, MA: Addison-Wesley) ch 8
[29] Good I J 1950 *Probability and the Weighing of Evidence* (London: Charles Griffin)
[30] Hubel D H and Wiesel T N 1977 The functional architecture of the macaque visual cortex. The Ferrier lecture *Proc. R. Soc.* B **198** 1–59
[31] Purves D, Riddle D R and LaMantia A-S 1992 Iterated patterns of brain circuitry (or how the cortex gets its spots) *TINS* **15** 362–8
[32] Geszti T and Pázmándi F 1987 Learning within bounds and dream sleep *J. Phys. A: Math. Gen.* **20** L1299–303
[33] Sandberg A, Petersson K M and Lansner A 2001 Selective enhancement of recall through plasticity modulation in an autoassociative memory *Neurocomputing* **38–40** 867–73
[34] Hertz J, Krogh A and Palmer R G 1991 *Introduction to the Theory of Neural Computation (Lecture Notes Santa Fe Institute for Studies in the Sciences of Complexity vol 1)* (Reading, MA: Addision-Wesley)
[35] Sejnowski T J 1989 Induction of synaptic plasticity by Hebbian covariance in the hippocampus *The Computing Neuron* ed R Durbin, C Miall and G Mitchison (Wokingham: Addison-Wesley)
[36] Bienenstock E L, Cooper L N and Munro P W 1982 Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex *J. Neurosci.* **2** 32–48
[37] Petersen C C, Malenka R C, Nicoll R A and Hopfield J J 1998 All-or-none potentiation at CA3-CA1 synapses *Proc. Natl Acad. Sci. USA* **95** 4732–7
[38] Andersen P, Sundberg S H, Sveen O, Swann J W and Wigström H 1980 Possible mechanisms for long-lasting potentiation of synaptic transmission in hippocampal slices from guinea-pigs *J. Physiol.* **302** 463–82
[39] Jester J M, Campbell L W and Sejnowski T J 1995 Associative EPSP-spike potentiation induced by pairing orthodromic and antidromic stimulation in rat hippocampal slices *J. Physiol.* **484** 689–705
[40] Bhalla U S and Iyengar R 1999 Emergent properties of networks of biological signaling pathways *Science* **283** 381–7
[41] Frey U and Morris R 1997 Synaptic tagging—synapse specificity during protein synthesis-dependent long-term potentiation *Nature* **385** 533–6
[42] Sternberg S 1966 High-speed scanning in human memory *Science* **153** 652–4
[43] Lisman J E and Idiart M A 1995 Storage of 7 ± 2 short-term memories in oscillatory subcycles *Science* **267** 1512–15
[44] Little W A and Shaw G L 1975 A statistical theory of short and long term memory *Behav. Biol.* **14** 115–33
[45] Brunel N, Carusi F and Fusi S 1998 Slow stochastic Hebbian learning of classes of stimuli in a recurrent neural network *Network* **9** 123–52
[46] Lansner A 1991 A recurrent Bayesian ANN capable of extracting prototypes from unlabeled and noisy examples *Proc. ICANN-91: Artificial Neural Networks (Espoo, Finland, June 1991)* ed T Kohonen, K Mäkisara, O Simula and J Kangas (Amsterdam: Elsevier) pp 247–54
[47] Squire L R 1992 Memory and the hippocampus: a synthesis from findings with rats, monkeys, and humans *Psychol. Rev.* **99** 195–231
[48] Wickens J and Kötter R 1995 Cellular models of reinforcement *Models of Information Processing in the Basal Ganglia* ed J C Houk, J L Davis and D G Beiser (Cambridge, MA: MIT Press) pp 187–214
[49] Woolf N J 1996 The critical role of cholinergic basal forebrain neurons in morphological change and memory encoding: a hypothesis *Neurobiol. Learn. Mem.* **66** 258–66
[50] Hasselmo M E, Wyble B P and Wallenstein G V 1996 Encoding and retrieval of episodic memories: role of cholinergic and GABAergic modulation in the hippocampus *Hippocampus* **6** 693–708