Proceedings of ALGORITMY 2000
Conference on Scientific Computing, pp. 314–319

NEW COMPUTER ALGORITHM ENABLING SIMULATION OF RECALL PROCESS IN SPARSELY ENCODED HOPFIELD-LIKE NEURAL NETWORK OF EXTREMELY LARGE SIZE *

FROLOV A.A. †, HÚSEK D. ‡, AND SNÁŠEL V. §

Abstract. The algorithm which minimizes the required computer memory to simulate neurodynamics in sparsely encoded Hopfield-like autoassociative memory is proposed. It does not require to keep in memory both connection matrix and a set of stored prototypes. The algorithm allows to simulate the recall process in the neural network of extremely large size and thus to calculate its asymptotic informational and dynamic properties when the number of network's nodes become infinite. Giving great advantage in computer memory the algorithm leads to some advantages in processing speed comparing with traditional algorithm when connection matrix and the set of learned prototypes are stored in computer memory.

Key words. Hopfield-like neural network, Hebbian rule

1. Introduction. Hopfield-like neural network is a fully connected network which acts as an autoassociative memory for statistically independent binary patterns on the basis of correlational Hebbian learning rule. Encoding is called sparse if the number of active neurons n in stored patterns (prototypes) is small compared with the total number of neurons in the network N. It was shown (Tsodyks & Feigel'man, 1988; Amari, 1989; Buhmann et al., 1989; Horner, 1989; Perez-Vicente & Amit, 1989, Frolov et al., 1991; Frolov et al., 1997 and others) that sparseness results in an increase of network capacity not only if it is measured by a number of stored prototypes (which would be a weak result because the entropy of each prototype decreases when becoming sparse), but it is valid even if *informational* capacity is considered which is measured by the total *entropy* of the stored prototypes. For example in the limit case for $p = n/N \to 0$ this entropy was estimated by Frolov et al. (1991) as $(N^2/2)\log_2 e \simeq 0.72N^2$, while for the original Hopfield network with $p = 0.5$ this entropy amounts to only about $0.14N^2$ (Hopfield, 1982; Amit et al., 1987). Thus sparsely encoded Hopfield-like network can be interesting even from the practical point of view. On the other hand it is useful as a model of some brain areas which are assumed to perform the functions of associative memory. Particularly it concerns CA3 hippocampal field (Rolls and Treves, 1997). Neural network activity in the brain is known to be rather sparse: p amounts to about 0.01 (Abeles, 1991).

Unfortunately all existing analytical methods fail being applied to sparsely encoded network (Frolov et al., in press). Thus the most reliable results concerning its informational and dynamic properties can be obtained only by computer simulation. However asymptotic behaviour of the network dynamics for $N \to \infty$ can be achieved only when simulated network is of extremely large size. Kohring's simulations (1990a,b) for nonsparse encoding ($p = 0.5$) showed that it is achieved when N is at least in the order of $10^4 - 10^5$. The same result has been obtained for sparse encoding (Frolov et al., in press). To perform the simulation of such large network

*This work was done under grant from the Grant Agency of Czech Republic, Prague No.: 201/97/0885 and Barrande Project No.: 99010-1

†Institute of Higher Nervous Activity and Neurophysiology Russian Academy of Sciences,
‡Institute of Computer Science Academy of Science of the Czech Republic,
§Department of Computer Science, Palacky University, Czech Republic

Kohring developed special algorithm which allows to avoid storing connection matrix in the computer memory but which, however, requires to store the whole set of prototypes. His algorithm is working only for nonsparse encoding. So we deduced new algorithm based on Kohring ideas and extended it such a way, that there is not necessary to store even larning prototypes. For sparse encoding this approach brings also other advantages. First, for nonsparse encoding the number of prototypes is much less then the network size and, second, components of prototypes are binary values while coefficients of connection matrix are integer values, it results in large economy of memory. On the basis of Kohring's approach we deduced appropriate formula for sparsely encoded network. Moreover we developed algorithm that allows for avoiding even storing the set of prototypes. Thus to simulate the network of a very large size one is restricted only by the computer time. Due to new algorithm we were able perform computer simulation even on a PC Pentium. The size of the network reached 10^5 neurons i.e it was in the same order as in simulations performed by Kohring with the use of Cray-YMP/832 computer.

2. Model description. As in (Frolov et al., 1997), we consider correlational Hebbian rule known to be much more efficient for network learning than the usual Hebbian rule (Buchmann at al., 1989; Tsodyks & Feigel'man, 1988). Hebbian rule is called correlational if components of the connection matrix are determined by equation

$$J_{ij} = \frac{1}{Np(1-p)} \sum_{l=1,L} (X_i^l - p)(X_j^l - p), i \neq j, \qquad J_{ii} = 0 \tag{2.1}$$

where $X_i^l \in \{0, 1\}$ are components of the prototypes (1 for active, 0 for nonactive neurons), $p = n/N$ is the probability for a neuron to be active and L is the total number of stored prototypes. In our model the prototypes are assumed to be uniformly and independently distributed within the pool of all the patterns with $n = pN$ component 1 and $N - n$ components 0 (i.e. the number of active neurons in each prototype is fixed). We showed in (Frolov et al., 1997) that this type of encoding provides an increase of informational capacity relative to the encoding when the number of active neurons in the prototypes is random and only its mean value is equal to n.

We restricted analysis on the case of parallel dynamics. Thus, the pattern of the network activity $\mathbf{X}$ at each time step t is calculated as

$$X_i(t+1) = \Theta(\eta_i(t) - T(t)), \qquad i = 1, ..., N \tag{2.2}$$

where

$$\eta_i(t) = \sum_j J_{ij} X_j(t) \tag{2.3}$$

is synaptic excitation, Θ is the step function and T is the activation threshold. The threshold $T(t)$ is chosen at each time step in such a way that the number of active neurons is equal to $n = pN$. So at each time step only n winners are firing. If several neurons have synaptic excitations equal to the activation threshold, then winners are the neurons with smaller index i. Since the number of active neurons in each prototype are fixed and also equal to n, this choice of activation threshold allows for a stabilization of network activity in the vicinity of one of the prototypes. Thus, the type of prototypes' encoding is fitted to the used simple recall procedure which allows to avoid explicit control of the activation threshold. This is the second advantage of the coding with fixed number of active neurons in prototypes. Our recall procedure also ensures that as in the case when activation threshold is fixed (Goles-Chacc et al., 1985) only two types of attractors (point or cyclic of the length two) are present in the network dynamics (Frolov et al., in press).

As a relative informational loading we use $\alpha = Lh(p)/N$ where $h(p) = -p\log_2 p - (1-p)\log_2(1-p)$ is the Shannon function. This measure of informational loading takes into account that an increase of sparseness results in a decrease of patterns' entropy, in contrast to the parameter α, defined as L/N, which has often been used as a measure of the network loading, since Hopfield's paper appeared (Hopfield, 1982). In the case of nonsparse encoding these two measures coincide because $h(0.5) = 1$. For sparse encoding we prefer to take into account patterns' entropy because the information capacity measured in this way approaches the finite value $\log_2 e/2$ when sparseness increases while information capacity defined as L/N diverges as $1/|p\log p|$.

3. Recall algorithm. To simulate the networks of a large size under the restriction of computer memory, we adjusted to the case of sparse encoding the procedure which has been suggested in Kohring (1990a) for nonsparse encoding. This procedure allows to simulate neurodynamics without storing the connection matrix and to express neurons' synaptic excitations through overlaps between the current pattern of the network activity and all stored prototypes. The overlap between the current pattern $\mathbf{X}(t)$ and one of the stored prototype $\mathbf{X}^l$ is defined as

$$m(\mathbf{X}^l, \mathbf{X}(t)) = \sum_{i=1,N} (X_i^l - p)X_i(t)/(Np(1-p)) \tag{3.1}$$

With the use of (2.1) and (2.3) one can easily obtain:

$$\begin{aligned}\eta_i(t) &= \frac{1}{Np(1-p)} \sum_{l=1,L} (X_i^l - p) \sum_{j\neq i} (X_j^l - p)X_j(t) \\ &= \sum_{l=1,L} ((X_i^l - p)m(\mathbf{X}^l, \mathbf{X}(t)) - \frac{1}{Np(1-p)}(X_i^l - p)^2 X_i(t)) \\ &= A_i - \frac{1-2p}{Np(1-p)}B_i - pC - \frac{LpX_i}{N(1-p)}\end{aligned} \tag{3.2}$$

where

$$A_i = \sum_{l=1,L} X_i^l m(\mathbf{X}^l, \mathbf{X}(t)), \qquad B_i = \sum_{l=1,L} X_i^l X_i(t)), \qquad C = \sum_{l=1,L} m(\mathbf{X}^l, \mathbf{X}(t)).$$

For sparse encoding it is more efficient to present stored prototypes as vectors of the length n whose components are indices of active neurons instead of vectors of the length N whose components are states (0 or 1) of all neurons. Then it is more convenient to represent the overlap between the current pattern of network activity and the stored prototype in the form

$$m(\mathbf{X}^l, \mathbf{X}(t)) = \frac{1}{Np(1-p)}(\sum_{i=1,N} X_i^l X_i(t) - Np^2) = \frac{1}{Np(1-p)}(\sum_{j=1,n} X_{i(l,j)}(t) - Np^2)$$

where $i(l,j), j = 1...n$ are indices of active neurons in l-th prototypes, i.e. components of the vector which represents this prototype. Thus only n additions are required to calculate each overlap and Ln additions to calculate all overlaps. Similarly for calculation of all N values A_i and B_i only $2Ln$ additions are required and approximately $3Ln$ additions are required to calculate all N values of synaptic excitations η_i in total. Note that for usual procedure based on the direct calculation of η_i by (2.3) nN additions are required. Thus for small loading the modified procedure has an advantage over the usual one even in the processing rate. Note also that for both procedures processing rate increases when sparseness increases.

The described procedure allows to avoid storing of the connection matrix but it requires to store the whole set of prototypes. However, the computer memory for storing the whole set of prototypes is rather large when N is large. In fact, to store about $N^2/2$ independent coefficients of the connection matrix as single-type variables, $2N^2$ memory bytes are required, while to store Ln indices of active neurons in prototypes as word-type variables, $2Ln$ bytes are required. For sparse coding we have $L \simeq \alpha N/|p \log_2 p|$. Thus, the memory which is required to store the set of prototypes is smaller only $|\log p|/\alpha$ times than that required to store the connection matrix. Computer simulation of the sparsely encoded Hopfield-like network is reasonable to perform for $\alpha \simeq 0.2 - 0.3$. Then the memory which is required to store the set of prototypes encoded with $p \sim 10^{-3}$ is approximately equal to about $0.05N^2$ bytes.

To avoid storing of prototypes we generated them at each recall step using the same sequence of pseudorandom values. These sequences are generated by pseudorandom generator (Sedgevick 1989) starting from the initial value characteristic for given sequence. To generate L patterns Ln pseudorandom values are required. To produce them approximately the same computer time is required as for calculation of synaptic excitations by eqn (3.2). Thus, we could avoid also the storing of the set of prototypes paying by the loss of processing rate in about only two times.

4. Results of computer simulation. To estimate the computation rate more accurately we performed computer simulation of the recall procedure with different values p, N and α on PC powered by Pentium processor. The program was written in Visual C++. The network size varied from 10000 to 50000 neurons and α varied from 0.001 to 0.1.

In agreement with the previous consideration the total time for computation of all N values of synaptic excitations η_i by eqn (3.2) (including computation of all overlaps and generation of all prototypes) was proportional to nL. The coefficient of proportionality τ_{mod} happened to be equal to about $2 \cdot 10^{-6}$ sec. The times for computation of one overlap and generation of one prototype were proportional to n. The corresponding coefficients of proportionality τ_{ov} and τ_{pr} happened to be equal to $0.5 \cdot 10^{-6}$ sec and $0.8 \cdot 10^{-6}$ sec respectively. Thus the time required to generate one prototype is only slightly larger than to compute one overlap.

Computer simulation of Hopfield network with the use of standard recall procedure given by eqn (2.3) has shown that in agreement with the previous consideration the time required to compute all N values of synaptic excitation is proportional to Nn. The coefficient of proportionality happened to be equal to $\tau_{st} \simeq 0.5 \cdot 10^{-6}$ sec. Thus to compute all synaptic excitations by standard algorithm one needs about $0.5Nn \cdot 10^{-6}$ sec. While to do the same by modified algorithm he needs about $2Ln \cdot 10^{-6}$ sec. For $L < N/4$ the modified algorithm has an advantage even in computing rate. Unfortunately this condition is reasonable only for computer simulation of densely encoded Hopfield network. For sparse encoding $L \simeq N\frac{\alpha}{p|log_2 p|}$, i.e for $\alpha \simeq 0.2 - 0.3$ $L >> N$ and the modified algorithm loses to standard one in computer rate.

It must be noted that for estimation of the rate of the standard algorithm we have not taken into account the time which is required for connection matrix computation. In the modified algorithm it is not necessary to compute the matrix at all. For standard algorithm this time is evidently proportional to N^2L. Computer simulation has shown that the coefficient of proportionality is equal to $\tau_{cm} \simeq 0.3 \cdot 10^{-6}$ sec. To estimate the properties of the recall procedure many testing trials are required for averaging. Usually the number of testing trials amounts to about $K = 100$ (Frolov et all, 1997). Thus to compute the connection matrix and then to estimate recall properties the total required time is equal to

$$T_{st} = \tau_{cm} * N^2L + \tau_{st}KSnN \tag{4.1}$$

where S is the mean number of time steps in the recall procedure. For

$$N > \frac{\tau_{st}KSph(p)}{\tau_{cm}} \tag{4.2}$$

the first term in eqn (4.1) dominates and the total testing time is mainly defined by the time for computing the connection matrix. The number of recall steps usually increases when N increases. However even for $N = 10^5$ it does not exceed 100 (Kohring, 1990b). For sparse encoding ($p << 1, \alpha \simeq 0.2 - 0.3$) condition (4.2) is valid under $N \simeq 10^5p^2 \log_2 p$. Even for $p = 0.1$ it is valid under $N \simeq 10^3$. Thus in fact for high sparseness the testing time for standard algorithm is completely defined by the time of connection matrix computation. Thus to compare the processing rates of modified and standard algorithms one must compare the total testing time $T_{st} = \tau_{cm}N^2L$ in standard algorithm with the total testing time $T_{mod} = \tau_{mod}KSnL$ in modified algorithm. The processing rate of modified algorithm is higher if $N > 18KSp$. Even for $p = 0.1$ this condition is valid for $N > 1.8 \cdot 10^4$. To reveal asymptotic behaviour of the network dynamics larger network sizes are required. Thus for high sparseness the suggested algorithm has an advantage over standard algorithm both in computer memory requirements and processing rate.

5. Conclusions. To reveal asymptotic behaviour of the network dynamics large network sizes are required. Straightforward implementation of simulation model is very memory demanding. Due to this fact there was no possibility to simulate large networks even on computers with relatively large operational memory. To solve this problem we developed the algorithm which minimizes the required computer memory to simulate neurodynamics in sparsely encoded Hopfield-like autoassociative memory. It does not require to keep in memory both connection matrix and a set of stored prototypes. The algorithm allows to simulate the recall process in the neural network of extremely large size and thus to calculate its asymptotic informational and dynamic properties when the number of network's nodes become infinite.

We have shown that proposed algorithm has not only great advantage regarding small memory footprint but it also has some advantages in processing speed comparing to traditional algorithm when connection matrix and the set of learned prototypes are stored in computer memory.

It must be noted that in spite of the relative advantage in computer rate the suggested algorithm requires a lot of computer time when the network size is actually extremely large. For example for $N = 10^5, p = 0.01, \alpha = 0.2$ the time required to compute all synaptic excitations (in fact to execute one step of the recall procedure) on PC Pentium-400 is equal to $\tau_{mod}Ln = \tau_{mod}N^2\alpha p/h(p) \simeq 5$ minute i.e. several hours are required until one recall process converges.

REFERENCES

[1] Abeles M.: Corticonics. Neural Circuits of the Cerebral Cortex, Cambridge, Cambridge University Press, 1991, 280 p. ISBN: 0-521-37617-3

[2] Amari, S.(1989). Characteristics of sparsely encode associative memory. *Neural Networks*, **2**, 451-457.

[3] Amit, D. J., Gutfreund, H., & Sompolinsky, H. (1987). Statistical mechanics of neural networks near saturation. *Annal of Physics*, **173**, 30-67.

[4] Buhmann, J., Divko, R., & Schulten, K. (1989). Associative memory with high information content. *Physical Review A*, **39**, 2689-2692.

[5] Frolov, A. A., Husek, D., & Muraviev, I. P. (1997). Informational capacity and recall quality in sparsely encoded Hopfield-like neural network: Analytical approaches and computer simulation. *Neural Networks*, **10**, 845-855.

[6] Frolov, A. A., Husek, D., & Muraviev, I. P. (in press). Informational efficiency of sparsely encoded Hopfield-like autoassociative memory. *Neural Networks.*

[7] Frolov, A. A., Mushinsky, A. M., & Tsodyks, M. V. (1991). Imitation model of associative memory as a neuron network with low activity level. *Biofizika*, **36**, 339-343.

[8] Goles-Chacc, E., Fogelman-Soulie, F., & Pellegrin, D. (1985). Decreasing energy functions as a tool for studying threshold networks. *Discrete mathematics*, **12**, 261-277.

[9] Hopfield, J. J. (1982). Neural network and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Science USA*, **79**, 2544-2548.

[10] Horner, H. (1989). Neural networks with low levels of activity: Ising vs. McCulloch-Pitts neurons. *Z Physik B*, **75**, 133-136.

[11] Kohring, G.A. (1990a). A high-precision study of the Hopfield model in the phase of broken replica symmetry. *Journal of Statistical Physics*, **59**, 1077-1086.

[12] Kohring, G.A. (1990b). Convergence time and finite size effects in neural networks. *Journal of Physics A: Math. Gen.*, **23**, 2237-2241.

[13] Perez-Vicente, C.J., & Amit, D.J. (1989). Optimized network for sparsely coded patterns Journal of Physics A: Math. Gen., **22**, 559-569.

[14] Rolls E.T., Treves A., Foster D., Perez-Vicente Comrado: Simulation Studies of the CA3 Hippocampal Subfield Modelled as an Attractor Neural Network, Neural Networks, Vol.10, 1997, No.9, pp.1559-1569

[15] Sedgewick R. Algorithms. Addison Wesley Publishing Company, 1989, pp. 657.

[16] Tsodyks, M.V. & Feigel'man, M.V. (1988). The enhanced storage capacity in neural network with low activity level. *Europhysical Letters*, **6**, 101-105.