Simulations with a large number of neurons

1989 J. Phys. A: Math. Gen. 22 L719

(http://iopscience.iop.org/0305-4470/22/14/012)

J. Phys. A: Math. Gen. 22 (1989) L719-L721. Printed in the UK

LETTER TO THE EDITOR

Simulations with a large number of neurons

T J P Penna and P M C Oliveira

Instituto de Física, Universidade Federal Fluminense, CP 100296, 24020, Niterói, RJ, Brazil

Received 25 May 1989

Abstract. We propose a new algorithm for evolution of neural networks, by division of the network into subnets, in order to reach retrieval of stored correlated patterns. We present a fast code for simulations with a large number of neurons (~10 000). The simulations were carried out on a microcomputer.

Since Hopfield's original paper (Hopfield 1982), where it was proposed that neural networks act like content-addressable memory, the retrieval and storage properties of neural networks have been intensively studied. The Hopfield model of neural network fails when the stored patterns are correlated, or when the neural network presents low levels of activity (Amit *et al* 1987a). Amit *et al* (1987b) used the replica-symmetry approach and numerical simulations considering up to 3000 fully connected neurons to obtain some important results concerning the Hopfield model. In this letter, we present a fast computer code for simulations with a large number of neurons, using both synchronous or asynchronous dynamics. We used up to 10 000 neurons, but this number may be increased. The simulations were carried out on an 80286-based microcomputer, running at 12 MHz. Our code spends ≃3 min/stored pattern in 10 000-neuron networks with 10^8 synapses, on the traditional Hopfield model. Each neuron holds an Ising spin (0 or 1) stored in only one bit of computer memory.

For the sake of simplicity we treated the zero-temperature case. However, the extension to finite temperatures is straightforward. The neuron states in an N-neuron network with P stored patterns are represented by $S_i = \pm 1$, and the coupling strengths are denoted by J_{ij}, as usual. The dynamics is governed by

$$S_i(t+1) = \operatorname{sgn}\left(\sum_{j=1}^{N} J_{ij} S_j(t)\right) \tag{1}$$

with

$$J_{ij} = \sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu \tag{2}$$

where $\xi_i^\mu = \pm 1$ represents the neuron i of the stored pattern μ. Equation (2) includes a self-interaction $J_{ii} = P$, which is usually set equal to zero, but can also be considered different from zero (Fontanari and Köberle 1988). One can easily consider any value J_{ii} by summing a term $(J_{ii} - P)S_i(t)$ inside the brackets of (1).

Usually, one evaluates the J_{ij} values, and stores them in an $N \times N$ matrix (the J_{ij} values are not altered during the dynamic process). In our computer code we do not store the values of J_{ij}, but only the patterns ξ^μ bit by bit on integer computer words.

0305-4470/89/140719+03$02.50

Therefore, we can use bitwise operations instead of the arithmetic operations appearing in (1) and (2), with no need to store partial results. The procedure is schematically as follows: we define new quantities: $\sigma_i = 1, 0$ corresponding to $S_i = -1, +1$ respectively and $\zeta_i^\mu = 1, 0$ corresponding to $\xi_i^\mu = -1, +1$, respectively. Then, the neuron states become

$$\sigma_i(t+1) = \frac{1}{NP}\left(2\sum_{j=1}^{N}\sum_{\mu=1}^{P}\zeta_i^\mu \otimes \zeta_j^\mu \otimes \sigma_j(t)\right) \tag{3}$$

where $\otimes$ corresponds to the XOR (eXclusive OR) logical operator and the final division by NP is of integer type. The usual $J_{ii} = 0$ case is obtained simply by substituting NP by $(N-1)P$ if $\sigma_i(t) = 0$ or by $(N+1)P$ if $\sigma_i(t) = 1$ in the final division; we have used this prescription in all simulations we have made. The trick we used is to interchange the order of summations. Inside the j summation, the product $\zeta_j^\mu \otimes \sigma_j$ is performed simultaneously for 16 neurons (in a 16-bit word processor) through only one XOR bitwise operation. The j summation is then performed simply by counting the number of bits 1 in the resulting word, through a POPCOUNT operation. This may be performed by a well defined routine using bitwise operations, but can be found, already implemented, on some versions of usual computer languages.

For uncorrelated patterns it is a well known fact that a discontinuous transition at $\alpha = \alpha_c \simeq 0.145$ exists, where $\alpha = P/N$. If we consider correlated patterns, α_c will be reduced drastically and no retrieval at all can be obtained in most cases. In order to test the behaviour of large networks we propose a new algorithm for the dynamics of the Hopfield model. We are interested in a neural network as a useful associative memory, not as a biological system. Consider, for example, two stored patterns with correlation x, defined as

$$x = 1 - \frac{2}{N}\sum_{i=1}^{N}\zeta_i^1 \otimes \zeta_i^2. \tag{4}$$

Hence, $x = 0$ means uncorrelated patterns and $x = +1$ (-1) means equal (opposite) patterns. If the two patterns have correlation x, they will have $N(1-x)/2$ 'different' neurons and $N(1+x)/2$ 'equal' neurons. Consider, for instance, $x > 0$. First we divide the network into two subnets: the first subnet has $N(1-x)$ neurons, where half of this number correspond to the different neurons, and the other half are chosen among the set of neurons which are equal in both patterns (any choice works). As a result the correlation between these two patterns inside this subnet is zero. The other subnet contains the Nx remainder neurons, all equal in both patterns. This means that for this subnet $x = 1$, and inside it both patterns can be considered as one. This approach can be extended to include more than two correlated patterns, or by division of the network into more subnets, or by a suitable choice of neurons that are equal in the two more correlated patterns, for only two subnets. The large number of neurons we can reach through our code is very important in this aspect. The next step is to let the two subnets evolve under the Hopfield dynamic independently, i.e. each subnet is fully connected, but there are no connections between distinct subnets. In table 1 we present the results obtained by taking one of the correlated patterns as input (without noise), for both the traditional Hopfield model and our model. Here, C is the mean correlation between the correlated stored patterns and m is the correlation between the pattern to be retrieved (the input) and its image after one complete update of the whole network ($T = 0$). Note that the value of m must be exactly 1 in this noiseless case: this is a necessary condition in order to reach retrieval when some initial noise

Table 1. Mean correlation between patterns (C) and correlation (m) between input pattern and its image after one complete upate, for the Hopfield model (m^a) and our model (m^b).

C	0.50	0.60	0.70	0.80	0.90
m^a	0.76	0.80	0.85	0.90	0.95
m^b	1	1	1	1	1

is included. We perform these simulations on a 6000-neuron network, with 10 uncorrelated and 50%-biased patterns and three (or four) correlated patterns. Thus we can only observe the effects due to pattern correlations, without saturation effects. The time needed to perform these simulations is reduced because the computer time involved in our simulations increases linearly with the number of stored patterns (P) and with the number of connections between neurons, and this last number is reduced after the subnet-division procedure. We believe that in a 32-bit processor the time will be considerably reduced. Our model is also robust for more than 30% of initial noise. We have also tested it for low level of activity patterns (5-10% of firing neurons). In our model the correlated part is a ferromagnetic fundamental state with unfiring neurons, for the low-level activity, and the final correlation is still 1. Generalisation to new rules like those of Gutfreund (1988) is straightforward.

The complete code was developed in C programming language (Kerningham and Ritchie 1978), for which there are the most efficient compilers for microcomputers, and is available on request from one of the authors. Translation to other languages is straightforward.

We would like to thaink R B Muniz for a critical reading of the manuscript. This work was partially supported by Brazilian agencies FINEP and CNPq.

References

Amit D J, Gutfreund H and Sompolinsky H 1987a *Phys. Rev.* A **35** 2293
—— 1987b *Ann. Phys., NY* **173** 30
Fontanari J F and Köberle R 1988 *J. Phys.: Math. Gen.* A **21** L667
Gutfreund H 1988 *Phys. Rev.* A **37** 570
Hopfield J J 1982 *Proc. Natl Acad. Sci. USA* **79** 2554
Kerningham B W and Ritchie D M 1978 *The C Programming Language* (Englewood Cliffs, NJ: Prentice-Hall)