5.0 References

- Ackley D., Hinton G. and Sejnowski, 1985,“A Learning Algorithm for Boltzmann Machines”, Cognitive Science, **9**, 147-169.
- Atick J. and Redlich A., 1990, “Towards a theory of early visual processing”, Neural Computation, **2**, 308-320.
- Barlow H., Kaushal T. and Mitchison G., 1989, “Finding Minimum Entropy Codes”, Neural Computation, **1**, 412-423.
- Becker S., 1992, “An Information-theoretic Unsupervised Learning Algorithm for neural networks”, Ph.D. Thesis, Univ. of Toronto.
- Bridle J., 1990, “Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters”, Neural Information Processing Systems, Vol. 2, 211-217, San Mateo, C.A. Morgan Kaufmann.
- Bridle J., MacKay D. and Heading A., 1991, “Unsupervised Classifiers, Mutual Information and ‘Phantom Targets’”, Neural Information Processing Systems, Vol. 4, 1096-1101, San Mateo, C.A. Morgan Kaufmann.
- Coomans D., Broeckaert M., Jonckheer M. and Massart D., 1983, “Comparison of Multivariate Discriminant Techniques for Clinical Data - Application to the Thyroid Functional State”, Meth. Inform. Med., **22**, 93-101.
- Földiak P., 1989, “Adaptive network for optimal linear feature extraction”. In Proc. IEEE/INNS Int. Joint Conf. Neural Networks, Washington, DC, Vol. 1, pp. 401-405. IEEE Press, New York.
- Glauber R., 1963, “Time -Dependent Statistics of the Ising Model”, Journal of Mathematical Physics, **4**, 294-307.
- Hentschel H.G.E. and Barlow H.B., “Minimum-entropy coding with Hopfield networks”, Network, **2**, 135-148.
- Herz J., Krogh A., Palmer R.G. 1990, “Introduction to The Theory of Neural Computation”, Addison-Wesley.
- Hopfield J.J., 1987, “Learning Algorithms and Probability Distribution in Feed-Forward and Feed-Back Networks”, Proc. of the National Academy of Sciences, USA 84, 8423-8433.
- Linsker R., 1988, “Self-organization in a perceptual network”, Computer, **21**, 105.
- Linsker R., 1989, “How to generate ordered maps by maximizing the mutual information between input and output signals”, Neural Computation**, 1**, 402-411.
- Linsker R., 1992, “Local Synaptic Learning Rules Suffice to Maximize Mutual Information in a Linear Network”, Neural Computation, **4**, 691-702.
- Schürmann B., 1989, “Stability and Adaptation in Artificial Neural Systems”, Physical Review A, **40**, n. 5, 2681-2688.
- Shannon C., 1948, “A mathematical theory of communication”, Bell System Technical Journal, **7**, 379-423.

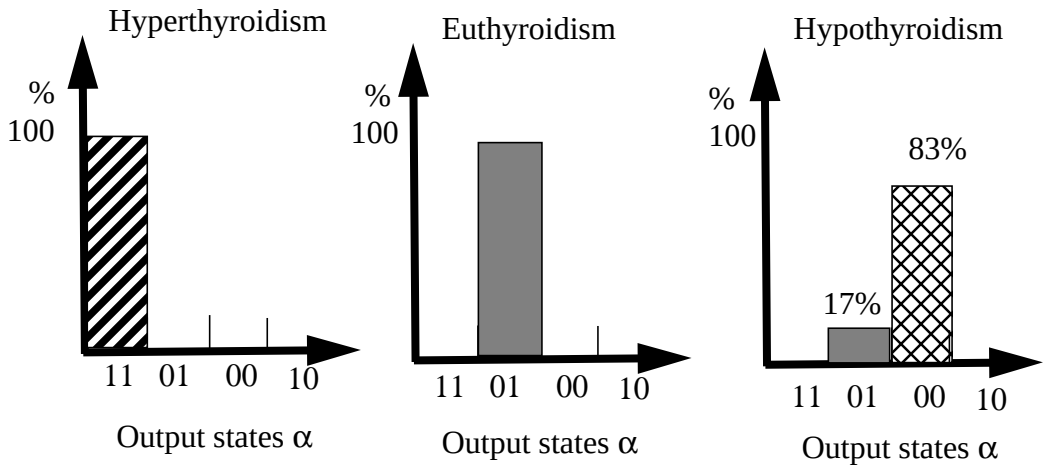

Figure 4

FIGURE 4. Graphical presentation of the correctness of classification after training. The overall performance of 95% correctness correspond to the perfect classification of two classes and 83% in the third. No data point falls in the possible output state 10 corresponding to the expert knowledge of three different possible diagnosis.

4.0 Conclusions

An unsupervised learning paradigm for a Boltzmann stochastic network was introduced in this paper. The proposed learning is an extension of the infomax principle for the case of a stochastic recurrent network. Maximum mutual information between the stochastic output neurons and the external inputs was used as learning criteria. The resulting learning rule consists of two terms corresponding to Hebbian and anti-Hebbian learning. The two terms are weighted by the amount of information transmitted in the learning synapse, giving and information theoretical interpretation for the proportionality constant of Hebb's biological rule. Simulations of the encoder problem and nonuniform binary inputs demonstrate the competitive performance of this method. Unsupervised classification of continuous inputs is shown for an artificial example and for a real world medical problem. An exponential smaller number of weights are required compared to known classification techniques.

FIGURE 3. Unsupervised classification of 4 Gaussian spots. The four different marks signalize the different output states of the tow output neurons. The evolution of the two weight vectors corresponding to the to output neurons are represented as a sequence of points.

3.4.2 Thyroid Diagnosis

In this section we present a real word clustering problem. Five labeled tests are used to learn to predict whether a patient's thyroid symptoms correspond to the three possible diagnosis: euthyroidism, hypothyroidism or hyperthyroidsim (Coomans et.al., 1983). The diagnosis (class label) was based on a complete medical record, including anamnesis, scan etc. The five inputs correspond to:

1) T3-resin uptake test.

2) Total Serum thyroxin.

3) Total Serum triiodothyronine.

4) Basal thyroid-stimulating hormone (TSH).

5) Maximal absolute difference of TSH value after injection of 200 micro grams of thyrotropin-releasing hormone as compared to the basal value.

There were 100 data samples for training. The stochastic network consists of five continuous inputs and two outputs neurons. After unsupervised class extraction three different classes were recognized, corresponding to the states 00, 01 and 11 of the output neurons. Note that we didn't use any class labels for training. The unsupervised extracted classes were compared with the real diagnosis after training yielding 95% correctness (see table 3). The unsupervised paradigm has extracted statistically correlated features of the input space corresponding to the true diagnosis. Note, that the unsupervised learning extracted only three states of the four possible output states (two neurons), which are in fact the three different possible diagnosis. Figure 4 shows the percentage of correct classification (relative to the number of elements of each tutor class) for the three different classes. The abscise represents the four possible output states of the network. Note, that two classes were correct classified to 100%. The output state 10 is not used. We have compared our results with a standard supervised back-propagation algorithm. The network structure consisted of five inputs, 15 hidden units and three outputs. The true classes were used during on-line learning. The obtained percentage of correct classification was 96%. We conclude, that obtained a remarkable result with the present unsupervised method.

We show in figure 3 the result of the unsupervised classification of four gaussian spots with equal variance. A network with two inputs (coordinates x1 and x2) and two outputs is used. The different classes correspond to the four possible states of the output layer, namely: 00, 10, 01, and 11. In figure 3 the four different marks correspond to the four different output states after training. The progress of the two weight vectors during training is shown as a sequence of points. Perpendicular lines to these weight vectors crossing the origin define the corresponding separation edges of the sigmoid function. The magnitude of the weight vectors determine the gain of the two sigmoids. The exact solutions are on the bisectors of the coordinate axes.

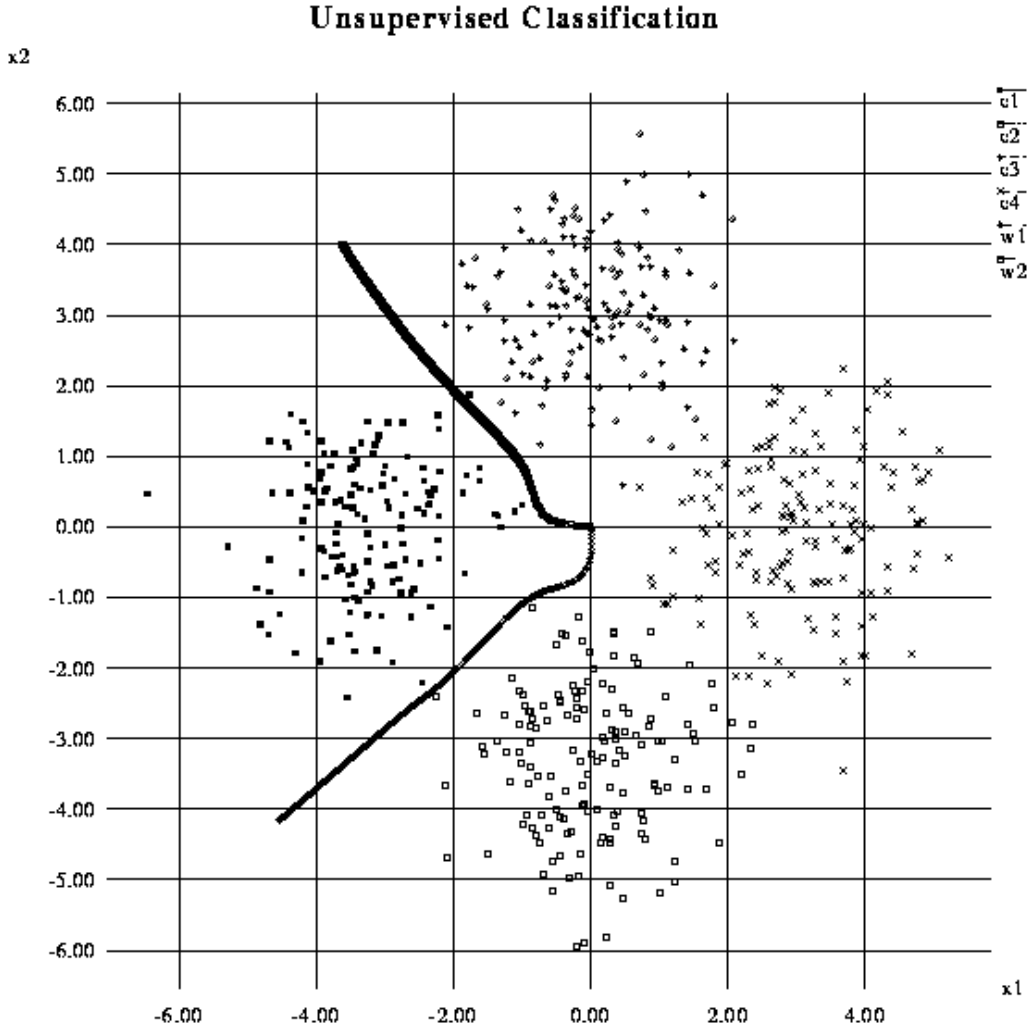

Figure 3

TABLE 2.

code I	code F
0 1 0 0 0 0 0 0	1 1 0
1 0 0 0 0 0 0 0	1 0 0

3.3.2 Power-law coding

As remarked by Hentschel and Barlow (1991) another important distribution is the power-law distribution $P_\gamma = K\gamma^{-2}$ with K being a proper normalization constant. This kind of distributions are of interest, since for example the word distribution in a normal English vocabulary follows this power-law distribution. The output layer has two neurons and the input code has again a local 4-dimensional code. All the results for the obtained revertible code are presented in table 4 and 5. In this case a code with reduced redundancy is found, but it is not exact factorial.

TABLE 3.

code	Bit Entropy	M	R
I	2.20	-	72%
F	1.28	1.28	5.47%

TABLE 4.

code I	code F
0 0 0 1	0 1
0 0 1 0	0 0
0 1 0 0	1 0
1 0 0 0	1 1

3.4 Unsupervised Classification

3.4.1 Gaussian Spots

The present model is also suitable for unsupervised classification of continuous valued inputs. The result can be interpreted as unsupervised “clustering” with sigmoid functions, which might be useful as preprocessing for supervised learning. Usually clustering is performed with gaussian functions or softmax functions (Bridle et.al., 1991). The most important difference is, that in our case the whole code expressed by the output layer is used as class label, reducing therefore the number of adaptive coefficients (see section 4.1). Usually each output (bit) corresponds to a class. In our model a state (word) of the output layer (labeled by α) represents a class.

The redundancy R measures the difference between the bit entropy $e(I)$ of code I and the initial entropy $H(\gamma)$ divided by $H(\gamma)$

$$R = \frac{e(I) - H(\gamma)}{H(\gamma)} \tag{3.4}$$

where the bit entropy is defined as,

$$e(I) = -\sum_i p_i \log(p_i) - \sum_i (1-p_i) \log(1-p_i) \tag{3.5}$$

with p_i being the probability of the ith bit taking the value 1 in the input code I. A vanishing redundancy indicates a factorial code.

3.3.1 Coding geometric progressions

It is possible to show that geometric progressions have an exact factorial representation consisting of a binary sequence coding (see Barlow et.al., 1989 and Hentschel and Barlow, 1991). We show in this section the results obtained using a input local representation of an 8-dimensional input forming a geometric progression $P_\gamma = Kx^\gamma$ with x=0.95 and K being a proper normalization constant. The output layer has three neurons. Table 5 shows the results after minimizing the mutual information. The code F is the invertible factorial code obtained after training. The condition of factorial code is not explicitly included in our learning rule. We obtained a factorial code by maximization of mutual information only because of the compression.

code	Bit Entropy	M	R
I	4.33	-	45%
F	2.99	2.99	0%

The code F is explicitly given in table 3.

TABLE 2.

code I	code F
0 0 0 0 0 0 0 1	0 1 0
0 0 0 0 0 0 1 0	1 1 1
0 0 0 0 0 1 0 0	0 1 1
0 0 0 0 1 0 0 0	0 0 1
0 0 0 1 0 0 0 0	0 0 0
0 0 1 0 0 0 0 0	1 0 1

TABLE 1.

input state	mapped to Output state
0 0 0 0 1	1 0 0
0 0 0 1 0	1 1 1
0 0 1 0 0	0 1 0
0 1 0 0 0	0 0 1
1 0 0 0 0	0 1 1

The same results were obtained using recurrent connections between the output neurons. Recurrences at the output layer can be interpreted as lateral inhibitions. The use of recurrences slightly accelerated the convergence to the maximum mutual information.

The experiment was repeated for the cases $4 \rightarrow 3$, $8 \rightarrow 3$, $40 \rightarrow 6$ and $5 \rightarrow 5$ with recurrent connections. In all cases perfect encoding was reached. When redundant number of output neurons are used, the mutual information criterion tries to decorrelated the neurons by generating a distributed code. This effect was also reported by Ackley et.al. (1985) for the supervised Boltzmann Machine.

The proposed unsupervised data compression outperform the results obtained with the conventional Boltzmann Machine. The maximum performance presented by Ackley et.al. (1985) was a $40 \rightarrow 10$ encoding problem with a 98.6% correct code. In the present model we were able to perform the maximum compression namely $40 \rightarrow 6$ with a perfect code (99.97% correctness) in a fully unsupervised way.

3.3 Compression of Non-uniform Distributions

The goal of the following examples is to find a nonlinear reversible transformation of a non-uniform distributed input code I in a compressed output code F. These kind of compression problems demonstrate empirically to be more difficult than the encoder problem. The reversibility assures complete information transmission into the new code F. The code I has a local input representation. In a local representation each input pattern has only one non-zero bit. Since it is a compression that maximize the transferred information, one expect that the most relevant information is extracted, i.e. the more representative features will be represented at the output. The most representative information at the output is expressed by a factorial code. A measure for this is the redundancy of the code. In the following subsections we will use the definition of redundancy as given by Barlow et.al. (1989).

FIGURE 1. Evolution of the mutual information to the global maximum values $\log_2(2)$ and $\log_2(3)$ for the $4 \rightarrow 2$ and $5 \rightarrow 3$ encoder problem respectively

For the $5 \rightarrow 3$ encoder problem the initial and final distributions after completion of the unsupervised learning are shown in figure 2. This figure shows the conditional probability distribution $P_{\alpha/\gamma}$ over the output states α for a given input pattern γ. Each curve describes the distribution for a different input. (Note that the abscess has only discrete values. The lines were drawn to visualize the results). At the beginning a uniform probability distribution is observed, due to the fact, that we start with small random weights ($[-0.001, 0.001]$). All codes are therefore equal probable, regardless of the given input. The mutual information is then zero. After training perfect binary data compression was obtained. This can be seen in figure 2, where for a given input patterns γ only one of the different codes α has $P_{\alpha/\gamma}$ equal to one and all others zero. For example the input code 1000 labeled by $\gamma = 1$ generates a distribution, which is equal to one at $\alpha = 3$ corresponding to the output state 001 and zero elsewhere. The codes α found at the output for the different input codes γ are presented in table 1.

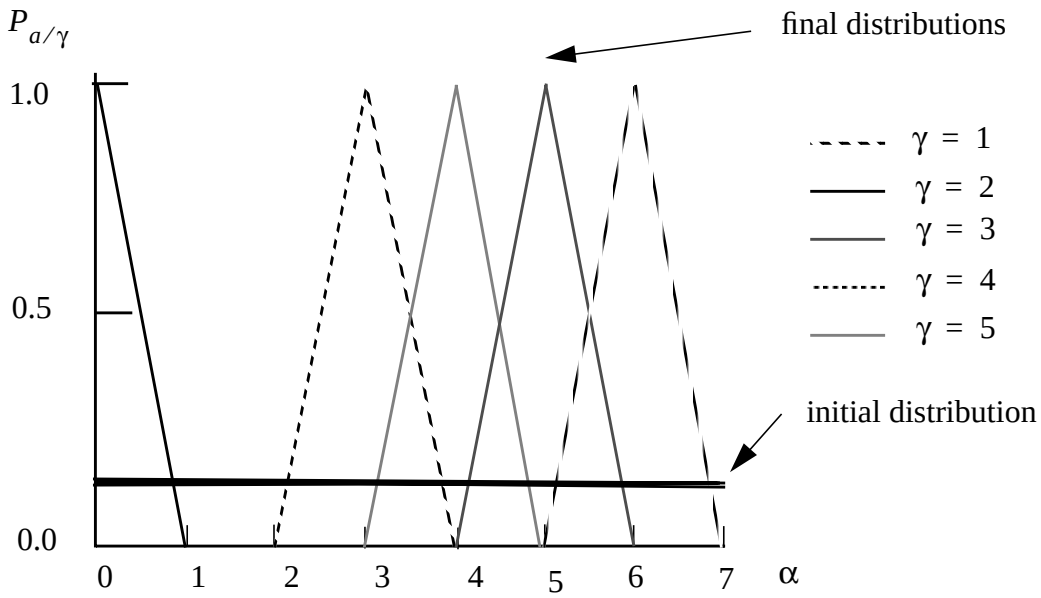

Figure 2

FIGURE 2. $.5 \rightarrow 3$ encoder problem.Final and initial probability distribution for the states of the output layer for the different patterns. The lines were drawn to facilitate the visibility of the results.Note that after training perfect binary data compression was obtained, since for different input patterns γ only one of the different codes α has $P_{\alpha/\gamma}$ equal to one and all others zero.

between the output units. For the examples presented in this paper network architectures without hidden units gave satisfactory results.

In the next section we demonstrate an application of the unsupervised Boltzmann network for binary data compression. Unsupervised classification of continuous input data is shown in the last subsection for an artificial and for a real world problem.

3.2 Encoder Problem

We test our learning paradigm with the same “encoder problem” as considered by Ackley et.al. (1985) for the original Boltzmann Machine. As the authors remarked, this problem can be understood as a simple abstraction of the recurring task of communicating information among various components of a parallel algorithm. The inputs for the $N \rightarrow n$ encoder, consist of N different patterns given by N input units, where only one unit has the value 1, and the others have the value 0. We assume a uniform pattern distribution ($P_\gamma = N^{-1}$). We use a network with n output neurons. In all cases, with the exception of the $5 \rightarrow 5$ encoding, recurrences in the output layer were disconnected. In all cases we used $T = 0.1$ and $\eta = 0.01$.

Figure1 shows the evolution of the mutual information to the global maximum values $\log_2(2)$ and $\log_2(3)$ for the $4 \rightarrow 2$ and $5 \rightarrow 3$ encoder problem respectively.

.

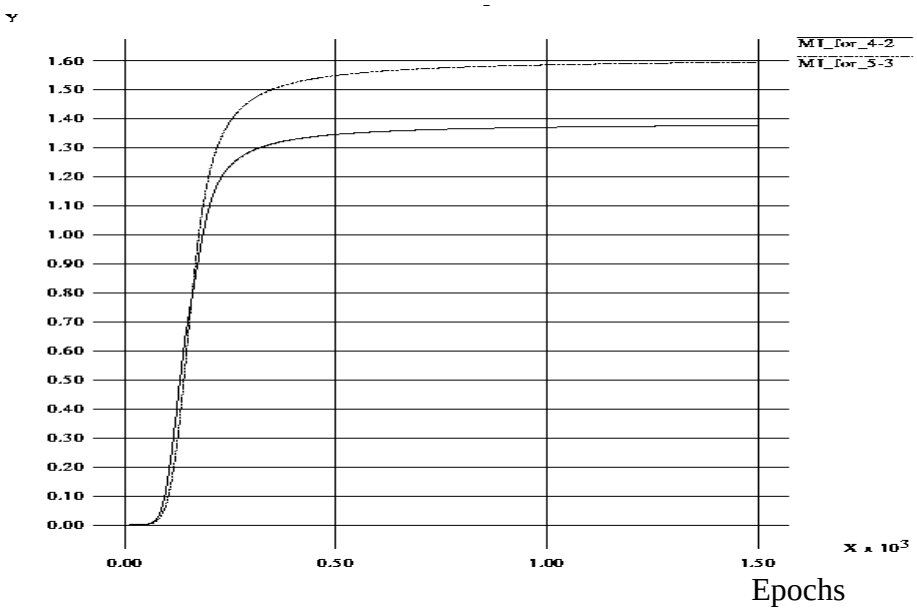

Figure 1

The Hebbian and anti-Hebbian terms are similar to the learning rule of the traditional Boltzmann Machine, but the calculated traces and averages differ considerably. In the traditional Boltzmann Machine the Hebbian and anti-Hebbian terms evaluate the differences between the “clamped” (supervised phase) and “free” phases. Remember that the “clamped” averaged in the Boltzmann Machine is done over the neuron states with the probability $R_{\alpha/\gamma}P_{\beta/\alpha\gamma}$, being $R_{\alpha/\gamma}$ the desired probability of the output units. The free or “free” averaged is done over the neuron states with the probability $P_{\alpha\beta/\gamma}$. In our model, we don’t have such phases due to the fact, that it is unsupervised. The Hebbian term in our case is a pure Hebbian without the artificial influences of “clamped” output states. The weighted sum in equation 2.11 and 2.12 are to be performed with the weighted probability $P_{\alpha\beta/\gamma}$. That means, that in the present learning paradigm the learning phases are always “free”. The Hebbian term appears for each possible state and the anti-Hebbian for the averaged correlation over all possible states. In fact equation 2.10 defines the same anti-Hebbian “free” average as in the traditional Boltzmann Machine.

3.0 Simulations and Results

3.1 Implementation and Complexity of the Model

We implemented the learning rule given by equation 2.13 and 2.14. It requires the calculations of probabilities $P_{\alpha/\gamma}$ with equations 2.2-2.4 and probabilities P_{α} by summing $P_{\alpha/\gamma}$ over the training patterns. We calculate this probabilities for all possible α, i.e. for the 2^n states of the output layer with n output neurons. Let us assume a d-dimensional input and N training patterns. The complexity of the algorithm in order to execute a single update of all weights is $O\left(d \cdot n \cdot N \cdot 2^n \right)$. It is the same as in the equivalent original Boltzmann Machine.

Let us analyze the case of a classification task for a d-dimensional input into p different classes. A maximum likelihood network or the unsupervised model of Bridle et al (1991) requires p output neurons and the complexity of the algorithm is $O(d \cdot p \cdot N)$ and the number of weights scale with $p \cdot d$. In our model the output layer should contain $log_2(p)$ units. The resulting complexity of the algorithm is some what higher $O(d \cdot log_2(p) \cdot N \cdot p)$. The advantage of this model for unsupervised classification task is not only the theoretical interpretation explained in the last section, but the fact, that it needs an exponentially smaller number of weights. In the present stochastic network the number of parameter would be $log_2(p) \cdot d$, which is to be compared with the number of weights of the traditional networks.

As pointed out already, this model and the learning rule allow for recurrent connections, and they apply as well to structures with canceled recurrent connections. In the following experiments we limited ourselves to one layer (output), but included recurrent connections

$$= \sum_{\gamma} P_{\gamma} \sum_{\alpha,\,\beta} \frac{\partial}{\partial w_{ij}} (P_{\alpha\beta/\gamma}) \log\left(\frac{P_{\alpha/\gamma}}{P_{\alpha}} \right) \tag{2.8}$$

In equation 2.7 the two last terms on the r.h.s. cancel each other, when the sum over γ in the third term is performed. Using equations 2.2, 2.3 and 2.4 we obtain

$$\frac{\partial}{\partial w_{ij}} (P_{\alpha\beta/\gamma}) = \frac{\tau}{2} P_{\alpha\beta/\gamma} \left(S_i^{\alpha\beta} S_j^{\alpha\beta} - \sum_{\alpha'\beta'} P_{\alpha'\beta'/\gamma} S_i^{\alpha'\beta'} S_j^{\alpha'\beta'} \right) \tag{2.9}$$

Introducing the average $\langle \ \rangle_{\gamma}$ over all free neurons states for a fixed input pattern γ,

$$\langle S_i S_j \rangle_{\gamma} = \sum_{\alpha'\beta'} P_{\alpha'\beta'/\gamma} S_i^{\alpha'\beta'} S_j^{\alpha'\beta'} \tag{2.10}$$

we can write the learning rule by combining equation 2.6-2.9 as

$$w_{ij}^{new} = w_{ij}^{old} + \frac{\tau\eta}{2} \cdot \sum_{\gamma} P_{\gamma} \sum_{\alpha,\,\beta} P_{\alpha\beta/\gamma} \log\left(\frac{P_{\alpha/\gamma}}{P_{\alpha}} \right) \cdot \left(S_i^{\alpha\beta} S_j^{\alpha\beta} - \langle S_i S_j \rangle_{\gamma} \right) \tag{2.11}$$

Mutatis mutandis, we obtain for the input connections the following learning rule,

$$W_{ij}^{new} = W_{ij}^{old} + \frac{\tau\eta}{2} \cdot \sum_{\gamma} P_{\gamma} \sum_{\alpha,\,\beta} P_{\alpha\beta/\gamma} \log\left(\frac{P_{\alpha/\gamma}}{P_{\alpha}} \right) \cdot \left(S_i^{\alpha\beta} X_j^{\gamma} - \langle S_i X_j^{\gamma} \rangle_{\gamma} \right) \tag{2.12}$$

If we cancel the hidden units the learning rule simplifies to

$$w_{ij}^{new} = w_{ij}^{old} + \frac{\tau\eta}{2} \cdot \sum_{\gamma} P_{\gamma} \sum_{\alpha} P_{\alpha/\gamma} \log\left(\frac{P_{\alpha/\gamma}}{P_{\alpha}} \right) \cdot \left(S_i^{\alpha} S_j^{\alpha} - \langle S_i S_j \rangle_{\gamma} \right) \tag{2.13}$$

$$W_{ij}^{new} = W_{ij}^{old} + \frac{\tau\eta}{2} \cdot \sum_{\gamma} P_{\gamma} \sum_{\alpha} P_{\alpha/\gamma} \log\left(\frac{P_{\alpha/\gamma}}{P_{\alpha}} \right) \cdot \left(S_i^{\alpha} X_j^{\gamma} - \langle S_i X_j^{\gamma} \rangle_{\gamma} \right) \tag{2.14}$$

The interpretation of the obtained unsupervised learning rule is interesting. A Hebbian term is given by $S_i^{\alpha\beta} S_j^{\alpha\beta}$ in equation 2.11. It gives the instantaneous correlation between the neurons. The second $-\langle S_i S_j \rangle_{\gamma}$ is an anti-Hebbian term given by the averaged correlation between the neurons. Both terms are weighted and summed over all possible states, being the weighting factor a measure for the information transmitted from one unit to the other in each possible state. Thus, we get a weighted correction according to whether the activation of the postsynaptic cell exceeds its average value or not. This considerations hold also for the connections between input and neurons.

measure for the transmitted information is given by the “mutual information” (Shannon 1948). In our case it can be written as

$$M = \sum_{\gamma} P_{\gamma} \sum_{\alpha} P_{\alpha/\gamma} \log\left(P_{\alpha/\gamma}\right) - \sum_{\gamma} P_{\gamma} \sum_{\alpha} P_{\alpha/\gamma} \log\left(\sum_{\gamma} P_{\gamma} P_{\alpha/\gamma}\right) \tag{2.5}$$

where P_{γ} is the probability distribution of the input patterns and $P_{\alpha/\gamma}$ is the conditional probability distribution of the output configurations given pattern γ at the input. Let us point out the differences to the traditional Boltzmann Machine. The traditional Boltzmann Machine perform “unsupervised” learning, since it is trained with an ensemble of external examples, in order to reproduce their probability distribution. The minimization of the cross-entropy between a given external distribution and the state distribution of the Boltzmann Machine leads to a density estimation of the environment. The aim of our model is to define an “unsupervised” learning, that extracts from the input environment the most representative “features”. A “feature”, corresponds to a strongly correlated group in the input space. Maximization of mutual information in the above stochastic network finds features which correspond to statistical correlation in the input space. These features or extracted classes are represented in the output code. The mutual information in this stochastic network is a measure of the information conveyed in the extracted classes. Certainly, this will be useful, only if the dimension of the output is lower than the dimension of the input. In this case the learning paradigm is expected to find the most representative features at the output, i.e. they contain the maximum amount of information about the input distribution. In the case, where the dimensions are equal, maximization of mutual information assures only reversibility of the extracted map, which means a mere copy of the eventually noisy input into the outputs. In our model the learning is completely unsupervised. We don’t have an external tutor (desired probability), and the only criterion used, is the maximization of transmitted information from input to output.

In order to maximize the mutual information we perform gradient ascendent corrections on the weights. This yields the following learning rule:

$$w_{ij}^{new} = w_{ij}^{old} + \eta \cdot \frac{\partial M}{\partial w_{ij}} \quad ; \qquad W_{ij}^{new} = W_{ij}^{old} + \eta \cdot \frac{\partial M}{\partial w_{ij}} \tag{2.6}$$

where η is a learning constant.

The derivatives of equation 2.6 can be calculated after some algebra, yielding

$$\frac{\partial M}{\partial w_{ij}} = \sum_{\gamma,\alpha,\beta} P_{\gamma}\left(\frac{\partial}{\partial w_{ij}}\left(P_{\alpha\beta/\gamma}\right)\right)\log\left(\frac{\sum_{\beta'} P_{\alpha\beta'/\gamma}}{P_{\alpha}}\right) + \sum_{\gamma,\alpha,\beta} P_{\gamma}\left(\frac{\partial}{\partial w_{ij}}\left(P_{\alpha\beta/\gamma}\right)\right) \tag{2.7}$$
$$- \sum_{\gamma,\alpha,\beta} P_{\gamma} P_{\alpha\beta/\gamma} / P_{\alpha} \sum_{\gamma',\beta'} \frac{\partial}{\partial w_{ij}}\left(P_{\alpha\beta'/\gamma'}\right) P_{\gamma'}$$

2.0 Theoretical Formulation

Let us define a neural network composed of stochastic binary units S_i, taking output value $S_i = 1$ with probability P_i and value $S_i = 0$ with probability $1 - P_i$. The probability P_i is given by

$$P_i = \frac{1}{1 + e^{-\tau\left(\sum_j w_{ij}S_j + \frac{1}{2}\sum_j W_{ij}X_j\right)}} \tag{2.1}$$

Stochastic units defined in this way describe the effect of thermal fluctuations in a system of Ising spins in the presence of an external field $h_i = \frac{1}{2}\sum_j w_{ij}x_j$. In the statistical physics this property describes Glauber dynamic (Glauber 1963). The parameter τ in equation 2.1 denotes an inverse temperature.

For symmetric connections w_{ij}, an energy function (Lyapunov function) can be defined, and the Boltzmann-Gibbs distribution gives the probability of finding the system in a specific state $\{S\}$. Lyapunov functions for asymmetric weights have been discussed by Schürmann (1989). The weights W_{ij} connect the external input vector $\underset{\sim}{X}$ with the net and need not to be symmetric. The external input vector $\underset{\sim}{X}^{\gamma}$ may assume real values drawn from a given probability distribution P_γ, with γ labeling the input patterns. Let us further label the state of the output units by α, and those of the hidden units by β. We point out that the input units should not be understood as Ising spins. For a concrete γ-pattern they are fixed, and determine a fixed external field. Then the Boltzmann-Gibbs distribution of the hidden and output states for a fixed input pattern γ can be written as

$$P_{\alpha\beta/\gamma} = \frac{e^{-\tau H^{\alpha\beta/\gamma}}}{Z_\gamma} \tag{2.2}$$

where Z_γ is the partition function

$$Z_\gamma = \sum_{\alpha\beta} e^{-\tau H^{\alpha\beta/\gamma}} \tag{2.3}$$

and $H^{\alpha\beta/\gamma}$ the energy function

$$H^{\alpha\beta/\gamma} = -\frac{1}{2}\sum_{ij} w_{ij} \cdot S_i^{\alpha\beta} \cdot S_j^{\alpha\beta} - \frac{1}{2}\sum_{ij} W_{ij} \cdot X_i^{\gamma} \cdot S_j^{\alpha\beta} \tag{2.4}$$

The unsupervised learning, introduced in this paper for a stochastic network, described by equation 2.2, consists in maximizing the transfer of information from the input vector to the output neurons. In other words, a message γ coded in the input vector should be transmitted through the stochastic neurons, so that the code given by the average thermal value of the output neurons contains the most information included in the original message γ. A

mann Machines can be seen as a generalization of Hopfield networks to include hidden units. The learning algorithm derived for these stochastic networks combines in an efficient way the relationship between Boltzmann distributions and information theory for density estimation. In a standard variation of the Boltzmann Machine density estimation is used for supervised training (Hopfield 1987, see also Herz et.al. 1990).

Simultaneously, information theoretical concepts were introduced by many authors (Becker, 1992, Bridle, 1990, Bridle et.al., 1991, Linsker, 1988,1989,1992) in order to characterize and model unsupervised learning techniques. Linsker (1988, 1989,1992) proposed an optimization principle, called “infomax”. According to it, synaptic weights adapt under boundary conditions in order to maximize mutual information between input and output layers of a cortical network. It has been proved (Atick and Redlich 1990) that statistically salient input features can be optimally extracted from noisy input by maximizing mutual information. Some algorithms were developed for maximizing mutual information by using probabilistic linear neurons (Linsker 1992) or non-linear neurons in a probabilistic “winner-take-all” network (Linsker 1989). In the linear case., the infomax principle is related to the Principal Component Analysis. Földiak (1989) demonstrates this by using deterministic networks (no noise on the output) and assuming decorrelated input noise.

The aim of the present work is to define an unsupervised learning paradigm for networks with Boltzmann Machine architecture. It is based on the maximization of the mutual information from the input space to a set of output neurons. This represents a nonlinear feature extraction from the environment space. In this way, we extend the infomax principle for probabilistic non-linear neurons, as well as for networks which include hidden neurons and recurrences. The learning algorithm yields an interesting weighted combination of Hebbian and anti-Hebbian rule. The weighted coefficients can be interpreted by the infomax principle. The algorithm is tested by using the encoder problem. Optimal data compression is obtained by using the algorithm presented herein. Unsupervised classification of even continuous inputs is possible. Bridle et.al. (1991) present the maximization of mutual information was as a criterion for unsupervised classification. The advantage of the present implementation of maximal mutual information is that the number of necessary weights for classification of a d-dimensional input in n classes is exponentially smaller. This is due to the fact that the networks used by Bridle et.al. (1991) need n outputs leading to dn weights. Instead, our model needs only $\log_2$n outputs and $d\log_2 n$ weights.

In section 2, the stochastic network architecture is defined and the mutual information between input and output is calculated. A learning rule for the maximization of the mutual information is derived and interpreted. Section3 presents simulations for the encoder problem and compression of nonuniform distributed binary inputs. The results of unsupervised classification of two examples with continuous inputs - Gaussian spots and the real world benchmark example of thyroid diagnosis classification - are presented in the last part of Section 3.

Unsupervised Learning for Boltzmann Machines

Gustavo Deco[1] and Lucas Parra[2]
[1,2] Siemens AG, Corporate Research and Development, ZFE ST SN 41
Otto-Hahn-Ring 6, 81730 Munich, Germany
[2]Ludwig-Maximilian-Universität München, Institut für Medizinische Optic
Gustavo.Deco@zfe.siemens.de

Abstract

An unsupervised learning algorithm for a stochastic recurrent neural network based on the Boltzmann Machine architecture is formulated in this paper. The maximization of the Mutual Information between the stochastic output neurons and the clamped inputs is used as an unsupervised criterion for training the network. The resulting learning rule contains two terms corresponding to Hebbian and anti-Hebbian learning. It is interesting that these two terms are weighted by the amount of information transmitted in the learning synapse, giving an information-theoretic interpretation of the proportionality constant of Hebb's biological rule. The anti-Hebbian term, which can be interpreted as a forgetting function, supports the optimal coding. In this way, optimal non-linear and recurrent implementation of data compression of boolean patterns are obtained. As an example, the encoder problem is simulated and trained in an unsupervised way in a one layer network. Compression of nonuniform distributed binary data is included. Unsupervised classification even for continuous inputs is shown for the cases of 4 overlapping gaussian spots and for a real world example of thyroid diagnosis. In comparison with other techniques, the present model requires for the classification problem an exponentially smaller number of weights.

1.0 Introduction

Boltzmann Machines (Ackley et.al.,1985) are a class of stochastic neural networks. The dynamic of these recurrent networks with symmetrical connections applies the principles of statistical mechanics, and is capable of learning a given probability distribution. Boltz-