A new online unsupervised learning rule for the BSB model

Conference Paper · February 2001

DOI: 10.1109/IJCNN.2001.939061 · Source: IEEE Xplore

2 authors:

Sylvain Chartier
University of Ottawa

Robert Proulx
Université du Québec à Montréal

A New Online Unsupervised Learning Rule for the BSB Model

Sylvain Chartier & Robert Proulx

Laboratoire d'études en intelligence naturelle et artificielle
Université du Québec à Montréal
Département de Psychologie
C.P. 8888 succ. Centre-ville
Montréal, Qc, H3C 3P8,Canada.
chartier@leina.uqam.ca proulx.robert@uqam.ca

Abstract

In this paper it is demonstrated that a new unsupervised learning rule enable non-linear model, like the BSB model and the Hopfield network, to learn online correlated stimuli. This rule stabilizes the weight matrix growth to the projection rule in a local fashion. The model has been tested with computer simulations that show that the model is stable over the variations of its free parameters and that it is noise tolerant in the recall task.

1. Introduction

The usefulness of unsupervised learning algorithms in artificial neural networks lie in their ability to naturally implement adaptive categorization without the need of postulating an access to pre-existing information from outside the system. Moreover, in a recurrent architecture those models are also able to categorize new exemplars from previously learned categories.

One example of such a model is the BSB neural network first introduced by Anderson et al [1]. This model, as any other neural network model, is completely specified by its architecture, its transmission rule and its learning rule.

2 The BSB neural network.

The BSB architecture is illustrated at the figure 1. It can be seen that the connections are autoassociative, in other words, a given stimulus vector is associated with it self.

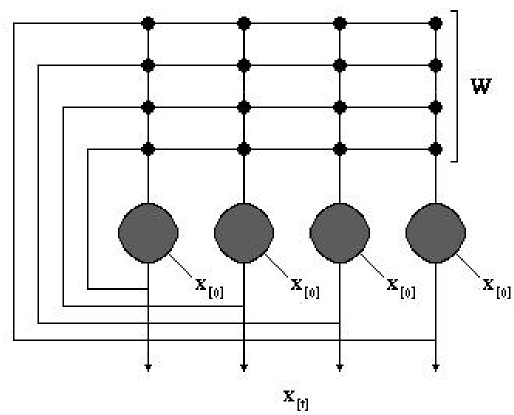

Figure 1: Illustration of the architecture of the BSB.

The transmission in this network is express by the following rule

$$\mathbf{x}_{[t+1]} = L[\mathbf{x}_{[t]} + \mathbf{W}\mathbf{x}_{[t]}],\ t = 1\ldots\mathrm{T} \tag{1}$$

Where $\mathbf{x}_{[t]}$ is the state vector that represent the activity of the units in the network at time t, **W**, the weight matrix and *L*[], a piecewise-linear function that constrains the activity of the units in an hypercube. More formally, the piecewise-linear function is express by

$$L[a] = \begin{cases} If \quad a > 1, \quad a = 1 \\ If \quad a < -1 \quad a = -1 \\ Else \quad a = a \end{cases} \tag{2}$$

The recurrence means that the states of the input vector influence its further states. Since the activity is hard limiting, after a finite number of cycles the state vector will stay in an attractor, a corner of the hypercube.

The learning rule consists of a simple hebbian learning formally stated by the following equation.

$$\mathbf{W}_{[k+1]} = \mathbf{W}_{[k]} + h(\mathbf{x}_{[7]}\mathbf{x}_{[7]}^{\mathrm{T}}) \quad (3)$$

Where, h represents a general learning parameter and the value k represents the learning trial. However, by using a simple hebbian learning rule the network is limited to learn only orthogonal stimuli without serious degradation in performance. Also, the use of a simple hebbian rule result in an exponential growth of the weight matrix that lead with the accumulation of learning trial in catastrophic large weight matrix values.

To overcome those difficulties many solutions have been propose (for examples see [2], [3]). One of these solutions is the Eidos model proposed by Bégin and Proulx [4].

2.1 The Eidos neural network

The Eidos model still use the same architecture and the same transmission rule previously introduced. However, this model uses a modified learning rule that enable the network to stabilize the growth of weight matrix and to learn correlated stimuli.

The learning rule in this model is formally expressed by

$$\Delta\mathbf{W}_{[k]} = \alpha\mathbf{x}_{[p]}\mathbf{x}_{[p]}^{\mathrm{T}} - \beta\mathbf{x}_{[n]}\mathbf{x}_{[n]}^{\mathrm{T}} \quad (4)$$

and

$$\mathbf{W}_{[k+1]} = \zeta\mathbf{W}_{[k]} + \Delta\mathbf{W}_{[k]} \quad (5)$$

Where α and β ($\alpha > \beta$) are general learning parameters, p and n ($p < n$) the number of iteration the state vector has to perform before the weight matrix is updated. The quantity ζ is a general forgetting parameter ($0 < \zeta \leq 1$).

The difference between the Eidos and the BSB model reside in the addition of a control factor, the anti-hebbian. It has been demonstrated [4] that provided the free parameters of the rule are set according to the previous constraint, the eigenvalues of the connection matrix will always converge to define the proper stimulus space, whatever the degree of correlation between the input stimuli.

Moreover, even after over-learning an old set of patterns, the presentation of a new linearly independent stimulus will always result in the emergence of a new positive eigenvalue in the spectrum and thus a new dimension in the feedback space.

However, this model is not exempt of problems. A first limitation is the incapacity of the model to learn online. Indeed, the necessary condition for the weight matrix to converge is that the transmission remains linear for all the learning trials. On the other hand, the interesting properties of the model reside in its non-linear transmission rule that keeps the state vector within the hypercube. Thus, the network can accomplish a learning or a recall task but not both at the same time. This context dependent use of the transmission rule restrains the network to learn online which limit its application in many engineering task and its biological plausibility.

A second limit of the learning rule is the constraint imposed on the iterations parameters p and n. Even if they are consider free, they have to be fix initially without the possibility to varied from a learning to trial to another. If they do, then the weight matrix will simply not converge.

A third limit is the use of two different general parameter α and β. Because α must be greater than β, this implies that a same units has two learning parameter used at different iteration time. Once again, this diminishes the robustness of the model.

Finally, before the patterns are presented to the network for learning they have to be normalized. Thus, a network construct on multiple layers of Eidos will necessitate the normalization of the state vector between each layer. This non-local process also diminishes the robustness of the model.

On the other hand, the weight matrix converges autonomously to the projection rule. Thus the solution found by the weight matrix is optimal according to least mean square error [5]. This solution enables the network to learn correlated patterns. These two facts advantage strongly the Eidos model over the BSB.

In the present research, we propose a novel learning rule that keeps all the interesting properties of the Eidos model without its limitations.

3 A new learning rule

The new learning algorithm is expressed by the following expression

$$\mathbf{W}_{k+1} = \mathbf{W}_{k} + h[\mathbf{x}_0\mathbf{x}_0^{\mathrm{T}} - (\mathbf{W}\mathbf{x}_t)(\mathbf{W}\mathbf{x}_t)^{\mathrm{T}}] \quad (6)$$

Where, h ($h > 0$) is a general learning parameter and t ($t > 0$) represents the number of iteration the state vector has to perform before the weight matrix is updated. Thus, there is only one general learning parameter and the number of iterations before making a weight update is also modulated by one parameter only.

This learning algorithm still holds an hebbian and a control factor part, the anti-hebbian. It can be interpreted as the difference between the forcing prototypes (hebbian part) minus the time-delayed feedback from the output of the network modulated by the weight connection matrix (anti-hebbian part).

However, before testing the new learning rule on a computer simulation task, we must show that this rule will self-stabilize over time.

3.1 Analysis on the new learning rule

The learning algorithm in the equation 6 can be decomposed for the learning of one eigenvalue. This decomposition is expressed by

$$\lambda_{k+1} = \lambda_k + h(1-\lambda_k^2) \tag{7}$$

If we take a closer look at this last equation we see that the variation of the eigenvalue is described by the quadratic function: $h(1-\lambda_k^2)$. This function is illustrated at figure 2.

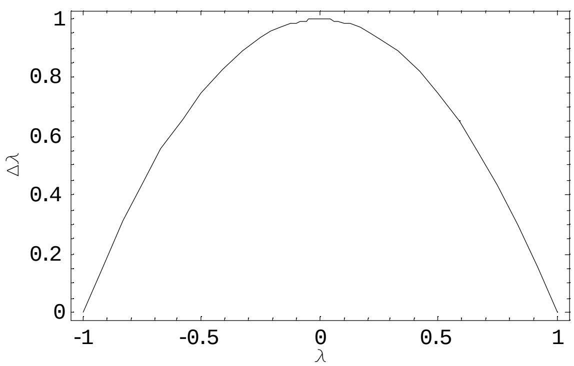

Figure 2: Graphics of the variation of the eigenvalue in function of its value, $\Delta\lambda_k = h(1-\lambda_k^2)$.

We can see that this function has two roots at –1 and 1. Thus the variation will be null at those values. However, the learning rule of the equation 6 is additive. Consequently, the weight matrix will self-stabilize once the eigenvalues has reached the value of 1. Still, we must insure that it does so without any oscillation.

To find the maximum value that the learning parameter can reach without any oscillation we must find the value for which the derivative of the function is less than zero. If we derive the equation 7 we obtain

$$\frac{d\lambda_{k+1}}{d\lambda_k} = 1 - 2h\lambda_k \tag{8}$$

Furthermore, if we set the value of λ_k to its converging value of 1, and make it less than zero we obtain

$$\frac{d\lambda_{k+1}}{d\lambda_k} = 1 - 2h < 0$$
$$h < \frac{1}{2} \tag{9}$$

This value is valid for one dimension only; however it can be generalized to D dimensions by setting the general parameter to a value

$$h < \frac{1}{2D} \tag{10}$$

Thus, if we set the value of the general learning parameter according to the relation expressed by equation (10), it is guarantee that all the eigenvalues of the weight matrix will converge to the value of 1.

Now we can turn to the question of what kind of final weight matrix result from the application of this learning rule. As it will be show, this learning algorithm also converges to the projection rule. To show this one can apply the learning algorithm to a prototype matrix. As previously demonstrated the weight matrix converge when its variation is null, thus it will be the case when

$$h(\mathbf{XX}^T - \mathbf{WX}(\mathbf{WX})^T) = 0$$
$$(\mathbf{X} - \mathbf{WX})(\mathbf{X} + \mathbf{WX})^T = 0 \tag{11}$$

If we keep only the left part of the left term we have

$$(\mathbf{X} - \mathbf{WX}) = 0 \tag{12}$$

This last equation has the following general solution [4]

$$\mathbf{W} = \mathbf{XX}^+ + \mathbf{Q}(\mathbf{I} - \mathbf{XX}^+) \tag{13}$$

Where, $\mathbf{Q}$, is an arbitrary matrix, $\mathbf{I}$, an identity matrix, and, $\mathbf{X}^+$ is the pseudo-inverse matrix. If we set $\mathbf{Q} = \mathbf{I}$, we obtain the trivial solution $\mathbf{W} = \mathbf{I}$, which is not an optimal solution for an autoassociative memory. However, if we set $\mathbf{Q} = \mathbf{0}$ we obtain

$$\mathbf{W} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T \tag{14}$$

This last equation gives the projection rule. Thus, the network will develop the proper subspace, whatever the correlation between the patterns. In other words, the resulting weight matrix is optimum in the least mean square sense. To support our analysis, the results of many computer simulations are presented in the next section.

3.2 Simulation I

The first simulation was to test if the network was able to learn online correlated patterns. The correlated patterns were alphanumeric letters placed on a 7x5 grid. Those letters can be seen at the figure 3. Each letter has been converted into a vector of 35 dimensions. From the image, a white pixel was given the value of -1 and a black pixel was given the value of +1. The correlation between the letters varies from 0.029 to 0.77. For the simulation we used the new learning rules expressed by the equation 5 and the BSB transmission rule express at the equation 1. For the simulation we set the learning parameter at the value 0.01 which is in agreement with the restriction expressed by the equation 10.

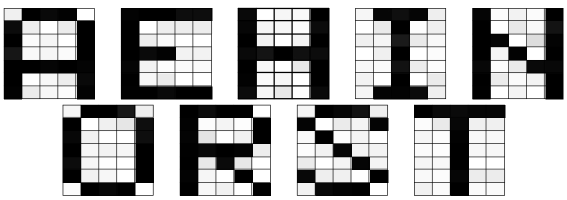

Figure 3: Graphics illustrating the patterns used for the simulations.

The simulation was done according to the following procedure.

1- Selection of a prototype at random.
2- Computation of $\mathbf{x}_t$ according of the BSB transmission rule (equation.1)
3- Computation of the weight matrix update according to the learning rule (equation. 6).
4- Repetition of 1 to 3 for 2000 learning trial.

We tested the BSB transmission rule in two conditions. In the first one, the number of iterations before the update of the weight matrix was fixed to t = 10. In the second, the number of iterations was varied randomly at each learning trial from t = 1 to 20.

After the learning has been accomplished we tested the network's performance on a recall task. The task consisted to categorize noisy version of the input patterns. The noisy version of the learning pattern was obtained by adding a random vector normally distributed with a mean of 0 and a standard deviation of π, where π represents the desired proportion of noise. This proportion has been varied from 10 to 100%. Figure 4 showed an example of a noisy version of the letter "S".

Each recall trial followed the following procedure

1- Selection of a prototype and addition of a new noisy vector.
2- Computation of $\mathbf{x}_t$ according to the transmission rule (equation 1).
3- Repetition of 2 until convergence of the sate vector.
4- Repetition of 1 to 3 for a different letter.

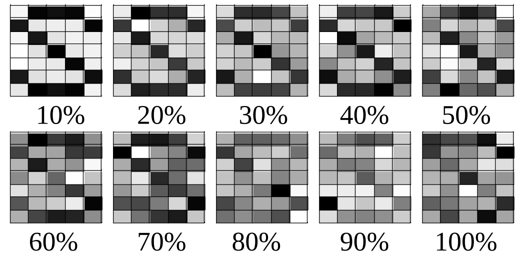

Figure 4: Graphics illustrating different proportion of noise for the letter "S".

To have a good idea of the performance of the network we presented 100 noisy version of a given letter. We did the same for the range of the desired proportion. From those manipulations we proceed to the simulations.

3.3 Results

The eigenvalue spectrums for each condition are presented in figure 5 below. Those results show that the number of iterations in the transmission rule does not influence the convergence of the weight matrix. In either case, fixed or random iterations, the network developed the same eigenvalue spectrum and the same final weight matrix.

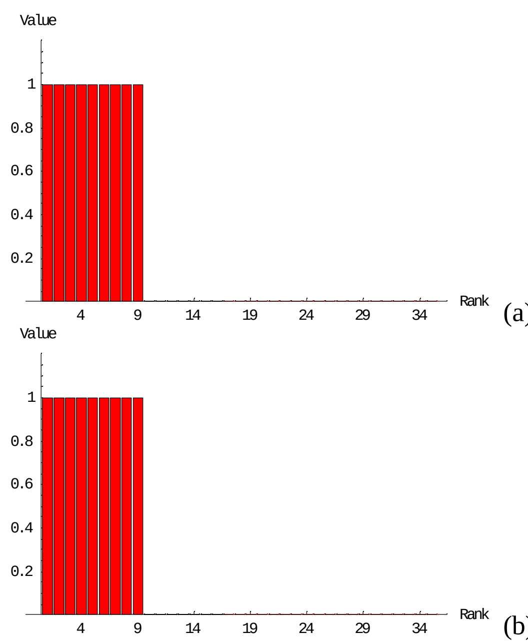

Figure 5: Graphics of the eigenvalues spectrum for the fixed (a) and the random condition (b).

Moreover, we can see from the graphics that the network develops the correct number of categories. Thus, every pattern has been adequately learned.

As for the recall task the figure 6 shows that the performance of the network start to decline around a noise of 90%. However even in the presence of a noisy vector composed of a proportion of 100% of noise, the performance remains acceptable at about 88%.

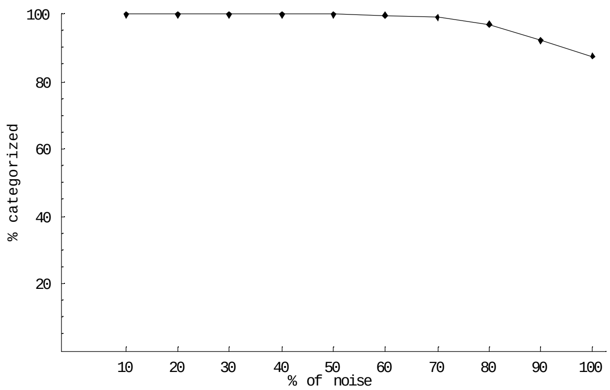

Figure 6: Graphic illustrating the percentage of correct categorized pattern under the variation of noise using the BSB transmission rule.

3.4 Discussion

The results presented above shows that the simple learning rule expressed at the equation 6 can achieve a good performance in the categorization of correlated patterns. This online rule does so with only one general learning parameter. Furthermore, the input pattern does not have to be normalized before it is presented to the network. We also see that the number of iterations in the transmission rule before the weight matrix is updated does not have an incidence on the final weight matrix.

As a conclusion, the new learning rule appear more efficient as well as more simple and more robust than any other earlier proposed learning algorithms.

From the previous results, it remains to see if the learning algorithm could be used with a more drastic nonlinear transmission rule? The next simulation provides an answer to that question.

3.5 Simulation II

In this simulation we used the same learning rule as the one described at the equation 6 but this time with an Hopfield activation function and the nonlinear bipolar signum function [6]. This transmission rule is express by the following equations.

$$\mathbf{x}_{[t+1]} = \operatorname{sgn}[\mathbf{W}\mathbf{x}_{[t]}],\ t = 1 \ldots \mathrm{T} \tag{15}$$

$$\operatorname{sgn}[a] = \begin{cases} \mathit{If} \quad a > 0, \quad a = 1 \\ \mathit{If} \quad a < 0 \quad a = -1 \\ \mathit{Else} \quad a = a \end{cases} \tag{16}$$

The simulation was conducted using the same methodology described previously.

The transmission rule was tested with a number of iterations fixed at t = 1. After the learning task, we measured the performance of the network in the same noisy categorization task.

3.6 Results

We can see from figure 7, that changing the transmission rule did not preclude the weight matrix to correctly self stabilize. Once again, the weight matrix develops the right number of eigenvalues.

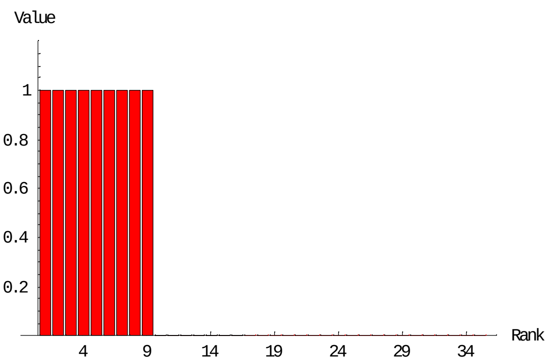

Figure 7: Graphic illustrating the eigenvalues spectrum of the weight matrix

Also, the performance of the model illustrated by the figure 8 indicates that the network still has a good performance. Its performance is roughly the same as with the BSB (see figure 6)

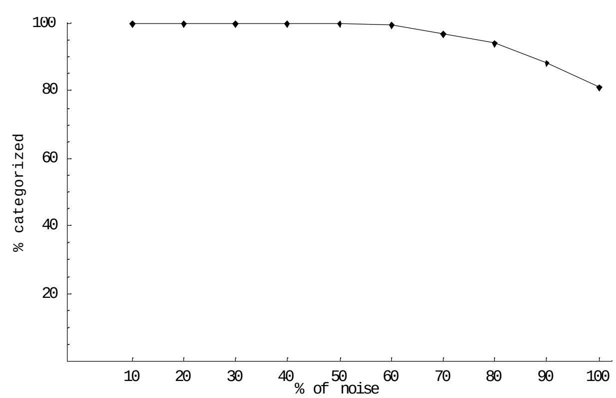

Figure 8: Graphic illustrating the percentage of correct categorized pattern under the variation of noise using the hopfield transmission rule.

3.7 Discussion

As expected from the preceding discussion the results clearly show that the learning rule can be applied to others type of transmission rule. Moreover, the results also demonstrate that the network is still able to stabilize under the presentation of correlated patterns even with an extreme non-linear transmission rule.

Also, because the learning rule converges to the projection rule, this rule can be use as an alternative to compute the weight matrix without using the pseudo-inverse, which implied the use of a non-local operation, as it is the case in Personnaz and Guyon [7].

Of course, further research will have to look if this learning rule can be generalized to continuous function.

4. Conclusion

From the previous results we have seen that the new unsupervised learning rule can be used to learn online correlated patterns. This rule stabilizes the weight matrix autonomously to the projection rule. We also showed that even under the variations of its free parameters the weight matrix still develop the right numbers of eigenvalues.

It is thus concluded that this learning rule possess sufficient robustness to constitute a general learning algorithm and can be used to increase the performance and the storage capacity of the usual correlational neural networks.

6. References

[1] Anderson, J. A. Silverstein, J. W. Ritz, S. A. and Jones, R. S. (1977). Distinctive features, categorized perception, and probability learning: some applications of a neural model. *Psychological review, 84*,413-451.

[2] Anderson, J.A., and Mozer, M., (1981). Categorization and Selective Neurons. In G. E. Hinton, & J. A. Anderson, (Eds.), *Parallel Models of Associative Memory* (pp. 213-236). Hillsdale, NJ: Erlbaum.

[3] Anderson, J. A. and Murphy, G. L., Psychological Concepts in a Parallel System. In J.D. Farmer, A. Lapedes, N. Packard, and B. Wendroff. (Eds.) *Evolution, Games, and Learning*. New York: North Holland, 1986.

[4] Bégin, J. and Proulx, R. (1996). Categorization in unsupervised neural networks: The Eidos model. *IEEE Transactions on neural networks, 7*, 147-154.

[5] Penrose, R. (1956). On best approximate solutions of linear matrix equations. *Proceeding of Cambridge philosophical society, 52*, 17-19.

[6] Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Science, E-U, 81*, 3088-3092.

[7] Personnaz, L., Guyon, L., and Dreyfus, G. (1985). Information storage and retrieval in spin-glass like neural networks. *Journal de Physique Lettre, 46*, L359-L365.