Learning in Multilayered Networks Used as Autoassociators *

M. Bianchini, P. Frasconi, and M. Gori, *Member, IEEE*

Dipartimento di Sistemi e Informatica, Universitá di Firenze
Via di Santa Marta 3 - 50139 Firenze - Italy
Tel. +39 (55) 479.6265 - Fax +39 (55) 479.6363
e-mail : marco@ingfi1.ing.unifi.it

Abstract

Gradient descent learning algorithms may get stuck in local minima, thus making the learning sub-optimal. In this paper, we focus attention on multilayered networks used as autoassociators and show some relationships with classical linear autoassociators. In addition, using the theoretical framework of our previous research [3], we derive a condition which is met at the end of the learning process and show that this condition has a very intriguing geometrical meaning in the pattern space.

Index Terms- Autoassociators, Backpropagation, multilayered networks.

I Introduction

It is quite a common opinion that in order to use multilayered networks successfully, a thorough understanding of the learning process is needed. Among the theoretical problems related to *Backpropagation* (BP), that of the error surface local minima is one of the most important. It has been investigated by numerous researchers in the attempt of finding examples of local minima and conditions for guaranteeing local minima free error surfaces. A general analysis on that problem, which also contains numerous references, can be found in [3], where some conditions are given for guaranteeing that the error surfaces are local minima free in the case of *pyramidal networks*. Roughly speaking the BP optimal convergence is guaranteed for networks with *many inputs* and *many hidden units*. In [3], the already cited analysis is specialized for the case of linearly separable patterns, which is likely to hold for patterns represented by "many coordinates". In [6, 7], the absence of local minima is guaranteed for networks with one hidden layer and as many hidden units as patterns. A similar condition leads obviously to nets with "many hidden units". In this paper, we propose an analysis of the learning for multilayered networks that turns out to be particularly useful for autoassociators. We discuss the role of the hidden neuron nonlinearity by showing that, in spite of linear autoassociators, any network configuration autoassociates patterns of a limited region. Concerning the problem of local minima however, we give an example that shows the

*This research was partially supported by MURST 40%.

problems introduced by the neuron nonlinearity and that supports some experimental failures reported in literature (see e.g. [2, 5]). The complex shape of the cost surface motivates accurate analyses for gaining more information on the learning process. Unfortunately, in the case of autoassociators the general analysis proposed in [3] does not hold, since it is conceived specifically for pyramidal networks. As for the studies by Yu [7] and Poston [6], they are likely to require a number of hidden units greater than the number of inputs (outputs), which contradicts the basic principle of autoassociators to compress the input information at the hidden layer. The study of the learned configurations proposed in this paper is based on the theoretical framework suggested in [3]. We give a condition that is necessarily met at the end of the learning process (*end-of-learning condition*). It holds in any case, no matter what number of hidden units is chosen, and with no relationship to the network interpolation capabilities. It involves the *output coordinate pattern correlation* and the *input-output coordinate pattern correlation* [1]. It is shown that, as a particular case, this condition has a very intriguing geometrical meaning in terms of *projection operators*. In the special case of linear hidden units, these operators become projection matrices and the proposed condition establishes the global optimization of the learning process.

II Notations and vectorial formulation

In this section, we define the formalism adopted throughout the paper. Basically, three entities need to be defined: a network $\mathcal{N}$, a learning environment $\mathcal{L}_e$ (set of data used for learning), and a cost index E_T.

- *Network* $\mathcal{N}$.
  It has a multilayered architecture. With reference to index l, we distinguish among the input layer ($l = 0$), the output layer ($l = L$), and hidden layers ($0 < l < L$). The number of neurons per layer is denoted by $n(l)$. Each neuron of layer l is referred to by its index $i(l)$: $i(l) = 1, \ldots, n(l)$. When pattern "t" is presented at the input, for each neuron, we consider $a_{i(l)}(t)$ (neuron $i(l)$'s activation), and $x_{i(l)}(t)$ (neuron $i(l)$'s output). The activation is computed by propagating forward the outputs of the previous level neurons as follows [2]:

$$a_{i(l)}(t) = w_{i(l)} \; + \sum_{j(l-1)=1}^{n(l-1)} w_{i(l),j(l-1)} x_{j(l-1)}(t), \tag{1}$$

  where $w_{i(l),j(l-1)}$ denotes the weight of the link between the neurons $j(l-1), i(l)$ and $w_{i(l)}$ is the threshold. In many cases, when using the network as an autoassociator, the output of neuron $i(l)$ is related to the activation by

$$x_{i(l)} = f(a_{i(l)}) \text{ for } l < L \text{ and } x_{i(L)} = a_{i(L)}, \tag{2}$$

  where $f(\cdot) : R \rightarrow [\underline{d}, \overline{d}]$ is a C^2 "squashing-like" function with a positive first derivative, matching the asymptotical conditions $\lim_{a \rightarrow +\infty} f(a) = \overline{d}$ and $\lim_{a \rightarrow -\infty} f(a) = \underline{d}$.

  Sometimes it is assumed that the outputs are also processed with a squashing function, that is $x_{i(L)} = f(a_{i(L)})$. In that case, the inputs must be properly preprocessed so as to be constrained in $[\underline{d}, \overline{d}]$, in order to impose the supervision required by the autoassociators.

[1] See section IV for a formal definition of these terms.

[2] In the sequel, both layer and pattern indices may be omitted for the sake of simplicity.

- *Learning Environment* $\mathcal{L}_e$.
  We use a set of supervised data for learning. In general, it is convenient to think of it as a collection of T input/output pairs for network $\mathcal{N}$. Since the network is used as an autoassociator, the targets $\mathcal{D}(t)$ are equal to the inputs, and therefore, the learning environment can be defined solely in terms of the inputs:
  $$\mathcal{L}_e \doteq \left\{ X_0(t) \in R^{n(0)}, t = 1, \ldots, T \right\}.$$
  $X_0(t)$ is the input pattern, also used as target.

- *Cost index.*
  For a given $\mathcal{L}_e$, the input-output data fitting is measured by means of the cost function
  $$E_T \doteq \frac{1}{2} \sum_{t=1}^{T} E_t = \frac{1}{2} \sum_{t=1}^{T} \sum_{j=1}^{n(L)} [x_{j(0)}(t) - x_{j(L)}(t)]^2. \tag{3}$$

In order to understand the basic result proposed in this paper, some more symbols must be introduced which allow us to deal with a compact vectorial formulation of the problem.

1. $X_{i(l)} \doteq [x_{i(l)}(1), \ldots, x_{i(l)}(T)]' \in R^T$ stores the output of neuron $i(l)$ for all the T patterns of the learning environment.

2. These vectors can be collected in the matrix $\mathcal{X}_l \doteq [X_{1(l)} \cdots X_{n(l)} \ \Pi] \in R^{T, n(l)+1}$, $0 \leq l \leq L$, where $\Pi \doteq [1, \ldots, 1]' \in R^T$ is used to deal with biases.

3. $w_{i(l), j(l-1)}$ is the weight which connects neuron $j(l-1)$ to neuron $i(l)$. The associated matrix $\mathcal{W}_{l-1} \in R^{n(l), n(l-1)}$ is referred to as *layer weight matrix.*

4. The *delta-error* is defined by $y_{i(l)}(t) \doteq \partial E_t / \partial a_{i(l)}(t)$, $Y_{i(l)} \doteq [y_{i(l)}(1), \ldots, y_{i(l)}(T)]'$, and $\mathcal{Y}_l \doteq [Y_{1(l)} \cdots Y_{n(l)}] \in R^{T, n(l)}$.

5. We assume that there is no connection which jumps a layer. Therefore, for weights connecting layers l and $l-1$ the gradient can be represented by a matrix $\mathcal{G}_{l-1} \in R^{n(l-1)+1, n(l)}$, whose generic element $g(j(l-1), i(l))$ is given by $\partial E_T / \partial w_{i(l), j(l-1)}$ if $j(l-1) \leq n(l-1)$, and $\partial E_T / \partial w_{i(l)}$ if $j(l-1) = n(l-1) + 1$ (bias term gradient contribution).

With these definitions the gradient matrix can be computed as follows [3]:

$$\mathcal{G}_{l-1} = \mathcal{X}'_{l-1} \mathcal{Y}_l, \qquad l = 1, \ldots, L. \tag{4}$$

III Linear versus nonlinear autoassociator

In this section, we discuss the relationships between linear and nonlinear autoassociators in order to give some general suggestions concerning the application to actual problems.

- **Nonlinear autoassociators perform pattern autoassociation of "clustered" regions**
  This property does not hold for linear autoassociators and consequently they can not be used for modeling patterns clustered somewhere in the space, since they only autoassociate linear spaces. In order to establish this concept more formally, let us define the following *ϵ-autoassociation* concept.

Definition
Let $\varepsilon \in \mathcal{R}^+$, $X_0(t)$, $X_L(t) \in R^{n(0)}$ be, where $X_L(t) = \mathcal{H}(X_0(t))$ and $\mathcal{H}$ is the mapping provided by the autoassociator. We say that $X_0(t)$ is ε-autoassociated if [3]

$$\frac{\| X_L(t) - X_0(t) \|_\infty}{\| X_L(t) \|_\infty} < \varepsilon. \tag{5}$$

Theorem 1 *Let us consider autoassociators with linear output units (see equation (2)). If the pattern $X_0(t)$ is ε-autoassociated, then $\forall\ \lambda : \lambda \geq \lambda_\varepsilon = (1+\varepsilon)||\mathcal{W}_{L-1}||_\infty / ||X_0(t)||_\infty$ the pattern $\lambda X_0(t)$ is not ε-autoassociated.*

Proof
Let us consider the following inequality:

$$\frac{\| \mathcal{H}(\lambda X_0(t)) - \lambda X_0(t) \|_\infty}{\| \mathcal{H}(\lambda X_0(t)) \|_\infty} > \varepsilon. \tag{6}$$

In order to prove the theorem we must show that there exists $\lambda_\varepsilon \in \mathcal{R}^+$ such that the previous inequality holds $\forall\ \lambda \geq \lambda_\varepsilon$. Since $||\mathcal{H}(\lambda X_0(t)) - \lambda X_0(t)||_\infty = ||\lambda X_0(t) - \mathcal{H}(\lambda X_0(t))||_\infty \geq ||\lambda X_0(t)||_\infty - ||\mathcal{H}(\lambda X_0(t))||_\infty$, the search of λ satisfying inequality (6) can take place also in

$$\lambda ||X_0(t)||_\infty \geq (1+\varepsilon)||\mathcal{H}(\lambda X_0(t))||_\infty. \tag{7}$$

If we find λ satisfying (7) then inequality (6) is automatically verified. Since the network has linear outputs, it is simply $X_L(t) = \mathcal{W}_{L-1} X_{L-1}(t)$. As a result

$$||X_L(t)||_\infty \leq ||\mathcal{W}_{L-1}||_\infty ||X_{L-1}(t)||_\infty \leq ||\mathcal{W}_{L-1}||_\infty. \tag{8}$$

From inequalities (7) and (8) it follows that $\lambda X_0(t)$ is not ε-autoassociated provided that $\lambda \geq \lambda_\varepsilon = (1+\varepsilon)||\mathcal{W}_{L-1}||_\infty / ||X_0(t)||_\infty$. ■

The meaning of this theoretical result is that only limited regions in the pattern space are ε-autoassociated, that is the nonlinear autoassociator with linear outputs performs a sort of *clustering* in the pattern space. Notice that, in the case of linear autoassociators, the previous property does not hold, since if $X_0(t)$ is ε-autoassociated then $\lambda X_0(t)$ is ε-autoassociated no matter what λ we choose.

A practical implication of this result is that nonlinear autoassociators with linear outputs turns out to be suitable for applications in which we need performing a clustering in the pattern space. For example, successful results have recently been found in the field of speech verification [4], where an autoassociator was used for modeling each speaker voice. When trained on the associated speaker voice, the nonlinear autoassociators performed very well in rejecting "false" speakers.

- **The cost function associated with nonlinear autoassociators can be populated by local minima**
  Baldi and Hornik [1] have proven that linear autoassociators produce local minima free error surfaces. The experimental feeling of many researchers however, is that this property does not hold

[3] We have chosen $||\ ||_\infty$ for simplicity. It is worth mentioning that the result remains valid for any equivalent norm, simply introducing the *ad hoc* constant in eq. (8).

when introducing nonlinearity in the hidden layer (see e.g. [2, 5]). Unfortunately, as in the case of pyramidal networks [3], the nonlinearity of the hidden units introduces local minima in the cost function that can make the learning very hard (see the Appendix).

IV The end-of-learning condition

Our analysis focuses on stationary points. We investigate what happens when the gradient of E_T is "zero", in order to discover what configurations cause learning algorithms to get stuck. A first general result can be derived by inspecting solely the null gradient condition w.r.t. the weights connecting the last layer. This result can be established for a network of any number of layers, but is reported here only for the case of one hidden layer, typically used in practice.

Theorem 2 *Let $\mathcal{N}$ be a network used as a linear output autoassociator for the learning environment $\mathcal{L}_e$. Under this assumption, the following condition* [4]

$$\mathcal{X}_2'\mathcal{X}_2 = \mathcal{X}_2'\mathcal{X}_0 \tag{9}$$

holds at the end of the learning.

Proof

At the end of the learning the gradient w.r.t. all the weights connected to the output layer must be null, $\mathcal{X}_1'\mathcal{Y}_2 = 0$, which, in turn, implies

$$\mathcal{W}_1\mathcal{X}_1'\mathcal{Y}_2 = \mathcal{X}_2'\mathcal{Y}_2 = 0. \tag{10}$$

Finally, since the outputs are not "squashed" and because of the Backpropagation relationship applied at the output layer, $\mathcal{Y}_2 = \mathcal{X}_2 - \mathcal{X}_0$, the thesis of the theorem follows:

$$\mathcal{X}_2'(\mathcal{X}_2 - \mathcal{X}_0) = \mathcal{X}_2'\mathcal{X}_2 - \mathcal{X}_2'\mathcal{X}_0 = 0. \tag{11}$$

■

At the end of the learning process, this condition imposes the equality of the *output coordinate correlation* $\mathcal{X}_2'\mathcal{X}_2$ and the *input-output coordinate correlation* $\mathcal{X}_2'\mathcal{X}_0$. In this paper, equation (9) is referred to as *end-of-learning* condition. Some remarks are worth mentioning.

Remark 1: If the network is capable of interpolating perfectly the given learning environment then the solution $\mathcal{X}_2 = \mathcal{X}_0$ obviously exists for (9). Moreover, if the same pattern $\tilde{t}$ is presented at the input T times ($rank\mathcal{X}_0 = 1$) equation (9) yields $Tx_{i(2)}^2(\tilde{t}) = Tx_{i(0)}(\tilde{t})x_{i(2)}(\tilde{t})\ \forall\ i(0) = i(2) = 1, \ldots, n(0)$, that implies $x_{i(2)}(\tilde{t}) = x_{i(0)}(\tilde{t})\ \forall\ i$, thus guaranteeing optimal learning. (Configuration in which some $x_{i(2)}(\tilde{t}) = 0$ are saddle points [5]. In the case of patterns clustered in the neighborhood of $\tilde{t}$ one may expect no dramatic changes for the cost function, particularly for "small clusters").

[4] Notice that the same result holds for any feedforward network with target $\mathcal{D}(t) \neq \mathcal{X}_0(t)$. In that case $\mathcal{X}_2'\mathcal{X}_2 = \mathcal{X}_2'\mathcal{D}$ holds at the end of the learning process.

[5] As it is easily verifiable in the analogous case of a pattern set containing only one pattern. In this case, for the 2–1–2 autoassociator (see fig. 1 in the Appendix), calculating the second derivative along $n = (n_1, n_2, n_3, n_4)'$, $\frac{d^2E}{dn^2}$, as an n_1-polynomial and supposing, for example, $x_{i(2)} = 0\ \ \forall\ i(2)$, the discriminant is $\frac{\Delta}{4} = f'^2(n_3x_{11}^2 + n_4x_{11}x_{12})^2 \geq 0$, thus assuring that the derivative changes in sign. Moreover such a point may be a zero for equation (9), but not for the gradient.

Remark 2: In order to understand the meaning of condition (9), let us take the trace on both sides and exchange the sums w.r.t. the coordinate i and the pattern t:

$$\sum_{t=1}^{T} \| X_2(t) \|^2 = \sum_{t=1}^{T} X_2'(t) X_0(t). \tag{12}$$

This relationship has a very intriguing geometrical meaning which comes out by considering the network transformation $\mathcal{H} : R^n \rightarrow R^n : \mathcal{H}(X_0(t)) \rightarrow X_2(t)$:

$$\sum_{t=1}^{T} \| \mathcal{H}(X_0(t)) \|^2 = \sum_{t=1}^{T} X_0'(t) \mathcal{H}(X_0(t)). \tag{13}$$

In the case of linear hidden units operator $\mathcal{H}$ is linear and the associated matrix H satisfies the relationship $H^2 = H$. Hence H is a *projection matrix* and represents the optimal solution. By analogy with Linear Algebra, operator $\mathcal{H}$ will be referred to as a *projection operator*. Equation (12) can also be given a further interesting interpretation. If we exploit the identity $\| X_2(t) - X_0(t) \|^2 = \| X_2(t) \|^2 - 2X_0'(t)X_2(t) + \| X_0(t) \|^2$, equation (13) becomes:

$$\sum_{t=1}^{T} \| X_0(t) \|^2 = \sum_{t=1}^{T} \| \delta(t) \|^2 + \sum_{t=1}^{T} \| X_2(t) \|^2, \tag{14}$$

being $\delta(t) \doteq X_0(t) - X_2(t)$. This equation makes it clear that the "efficiency" of the learning depends on "energy" $\sum_{t=1}^{T} \| \delta(t) \|^2$. Since matrix $\mathcal{X}_2 \in R^{T, n(2)}$ has rank $n(1)$, at most, for an efficient learning, a dimensionality-reduction of the input pattern must be possible.

Remark 3: Equation (12) can be rewritten for fully nonlinear autoassociators (with nonlinearity even at the output layer). In this case, by imposing the null gradient condition $g_{i(2),j(1)} = 0$, the following condition follows

$$g_{i(2),j(1)} = \sum_t x_{j(1)} y_{i(2)} = \sum_t x_{j(1)} x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) = 0, \tag{15}$$

since for squashing function f, $f' = f(1-f)$ holds. If we right-multiply $w_{i(2),j(1)}$ and sum up w.r.t. $j(1)$ we obtain

$$\begin{aligned} \textstyle\sum_{j(1)} w_{i(2),j(1)} g_{i(2),j(1)} &= \textstyle\sum_t (\sum_{j(1)} w_{i(2),j(1)} x_{j(1)}) x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t a_{i(2)} x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t (\log \frac{x_{i(2)}}{1 - x_{i(2)}}) x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t \Phi(x_{i(2)})[x_{i(2)} - x_{i(0)}] = 0, \end{aligned} \tag{16}$$

being $\Phi(x) \doteq x(1-x) \log \frac{x}{1-x}$. Summing up over $i(2)$ yields

$$\sum_t \sum_{i(2)} \Phi(x_{i(2)})[x_{i(2)} - x_{i(0)}] = 0. \tag{17}$$

This equation is the natural generalization of the projection theorem stated by (12). Since $lim_{x \rightarrow 1} \Phi(x) = \Phi(1/2) = 0$, equation (17) suggests that, unlike what happens in the case of linear outputs, patterns with values close to 0.5 and 1 do not contribute to the end-of-learning condition.

Remark 4: It is worth mentioning that the *end-of-learning* relation was obtained only by exploiting the condition of null gradient on stationary points w.r.t. the output layer. Let us consider the implications of imposing the null gradient at the hidden layer. For nonlinear autoassociator, we cannot write the global null gradient condition, although for neuron $i(1)$ $\mathcal{X}_0'\mathcal{D}_{i(1)}(\mathcal{X}_2 - \mathcal{X}_0)W_{i(1)} = 0$ holds, where $\mathcal{D}_{i(1)} = diag(f'(a_{i(1)}))$. Finally, right-multiplying by $\mathcal{X}_1'$ yields

$$\widehat{\mathcal{X}}_{0_{i(1)}}'(\mathcal{X}_2 - \mathcal{X}_0)X_{2_{i(1)}}' = 0, \tag{18}$$

being $\widehat{\mathcal{X}}_{0_{i(1)}}'$ a *local-scaled* version of $\mathcal{X}_0'$. For linear autoassociators equation (18) becomes $(\mathcal{X}_2'\mathcal{X}_2 - \mathcal{X}_0'\mathcal{X}_0)\mathcal{X}_2' = 0$. Finally, in the *linear symmetrical* case, $(\mathcal{W} \doteq \mathcal{W}_0 = \mathcal{W}_1')$, the null gradient condition on output layer is $\mathcal{X}_1'\mathcal{Y}_2 = 0$, or equivalently $\mathcal{W}_0\mathcal{X}_0'\mathcal{Y}_2 = 0$, while, $\mathcal{G}_0 = 0$ yields $\mathcal{X}_0'\mathcal{Y}_1 = \mathcal{X}_0'\mathcal{Y}_2\mathcal{W}_1 = 0$. Then matrices $\mathcal{X}_0'\mathcal{Y}_2$ and $\mathcal{Y}_2'\mathcal{X}_0$ must belong to the kernel of matrix $\mathcal{W}$. Because of the BP relationship on the output layer $\mathcal{X}_0'\mathcal{Y}_2 = \mathcal{X}_0'(\mathcal{X}_2 - \mathcal{X}_0) = \mathcal{X}_0'\mathcal{W}'\mathcal{W}\mathcal{X}_0 - \mathcal{X}_0'\mathcal{X}_0$ is a symmetric matrix. As a consequence $\mathcal{G}_0 = \mathcal{X}_0'\mathcal{Y}_2\mathcal{W}_1 = 0$ implies $\mathcal{W}_1'\mathcal{Y}_2'\mathcal{X}_0 = \mathcal{W}_0\mathcal{X}_0'\mathcal{Y}_2 = \mathcal{G}_1 = 0$ and, therefore, in this case, equation (9) define completely all the stationary points.

V Conclusions

This paper discusses on the relationships between linear and nonlinear autoassociators. It turns out that the better computational capabilities of nonlinear autoassociators can be very useful in practice, but, unfortunately, their learning can be seriously plagued by local minima in the error surface.
This paper gives some insights for better understanding the configurations "magically" learned by algorithms like Backpropagation. A condition is given that must necessarily be met at the end of the learning, no matter what algorithm is used. This condition has a very intriguing geometrical meaning in the pattern space.

Acknowledgements
We would like to thank Marco Maggini for his very useful comments and suggestions.

APPENDIX

Suboptimal Learning in Nonlinear Autoassociators

Let us consider the network depicted in fig. 1 and suppose it is fed by

$$\mathcal{X}_0 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}. \tag{19}$$

As $rank\mathcal{X}_2$ is at most 1, suppose a possible solution to the autoassociation problem is

$$\mathcal{X}_2 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \tag{20}$$

where the first pattern is mismatched to the pair $(0,0)$, whereas the second one is exactly autoassociated. In the case of two patterns, cost function E can be written as

$$\begin{aligned} E = \quad \tfrac{1}{2} \quad & \{[f(x_{11}w_{31} + x_{12}w_{32})w_{43} - x_{11}]^2 + [f(x_{11}w_{31} + x_{12}w_{32})w_{53} - x_{12}]^2 + \\ & [f(x_{21}w_{31} + x_{22}w_{32})w_{43} - x_{21}]^2 + [f(x_{21}w_{31} + x_{22}w_{32})w_{53} - x_{22}]^2\}, \end{aligned} \tag{21}$$

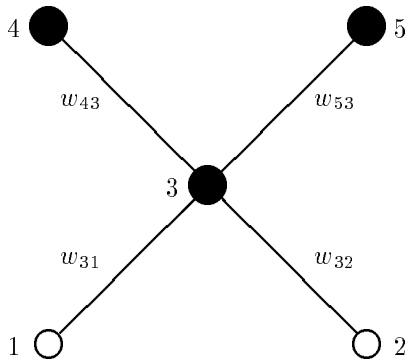

Figure 1: Autoassociator 2–1–2.

where function f is the identity in the case of linear autoassociator, otherwise $f = \frac{1}{1+e^{-x}}$. Thus, the associated gradient vector is:

$$\nabla E = \begin{bmatrix} \sum_{i=1}^{2} f'(a_{3i}) x_{i1} [(x_{3i} w_{43} - x_{i1}) w_{43} + (x_{3i} w_{53} - x_{i2}) w_{53}] \\ \sum_{i=1}^{2} f'(a_{3i}) x_{i2} [(x_{3i} w_{43} - x_{i1}) w_{43} + (x_{3i} w_{53} - x_{i2}) w_{53}] \\ \sum_{i=1}^{2} [x_{3i} w_{43} - x_{i1}] x_{3i} \\ \sum_{i=1}^{2} [x_{3i} w_{53} - x_{i2}] x_{3i} \end{bmatrix}, \tag{22}$$

being $a_{3i} = x_{i1} w_{31} + x_{i2} w_{32}$, $x_{3i} = f(a_{3i})$ and $f'(a_{3i}) = 1$ for linear autoassociator. As can be proven by substitution in equation (22), $\mathcal{X}_2$ is not a solution in the linear case. Forcing the null gradient condition yields $x_{31} = 0$ from the last two components and $w_{43} = -w_{53}$ from the first. Being $w_{43} \neq 0$, from $x_{32} w_{43} = 1$ $x_{32} = \frac{1}{w_{43}}$ follows. As a consequence, $x_{32} w_{53} = -\frac{1}{w_{43}} w_{43} = -1 = 0$, which is impossible.
In the nonlinear autoassociator the null gradient condition on the last two components produces again $x_{31} = 0$ [6], which, in turn, implies $f'(a_{31}) = 0$, $f''(a_{31}) = 0$, and so on. Moreover $x_{31} = 0$ also guarantees that the first two components of the gradient are null. As a consequence $\mathcal{X}_2$ is a stationary point. The Hessian matrix evaluated at this point

$$H = \begin{bmatrix} f'^2(a_{32})(w_{43}^2 + w_{53}^2) & 0 & f'(a_{32}) & 0 \\ 0 & 0 & 0 & 0 \\ f'(a_{32}) & 0 & x_{32}^2 & 0 \\ 0 & 0 & 0 & x_{32}^2 \end{bmatrix} \tag{23}$$

has null determinant. Nevertheless, writing the expression for the related second derivative along $n = (n_1, n_2, n_3, n_4)'$ yields

$$\frac{d^2 E}{dn^2} = n' H n = n_1^2 [f'^2(a_{32})(w_{43}^2 + w_{53}^2)] + 2 n_1 [n_3 f'(a_{32})] + [(n_3^2 + n_4^2) x_{32}^2], \tag{24}$$

which may be interpreted as a second degree polynomial on n_1 with constant positive sign, as $\frac{\Delta}{4} = -n_4^2 f'^2(a_{32})$. Therefore $\mathcal{X}_2$ is the output associated with a "*local minima configuration*", being $(0, 1, 0, 0)$, the direction corresponding to the infinite weight w_{32}, the only one for which $\frac{d^2 E}{dn^2}$ is null. (So there is a closed canyon with only one flat way out, which, numerically, acts as a minimum).

[6] In this case, $x_{31} = 0$ is simply accomplished choosing arbitrarily $w_{31} \neq \infty$ while $w_{32} \to -\infty$.

References

[1] P. Baldi and K. Hornik, "Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima," *Neural Networks*, vol. 2, 1989, pp. 53-58.

[2] H. Bourlard and Y. Kamp, "Auto-Association by Multilayer Perceptrons and Singular Value Decomposition," *Biological Cybernetics*, n. 59, 1988, pp. 291-294.

[3] M. Gori and A. Tesi, "On the problem of local minima in Backpropagation," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-14, no. 1, January 1992, pp. 76-86.

[4] L. Lastrucci, M. Gori, and G. Soda, "Neural Autoassociators for Phoneme-Based Speaker Verification," *Proc. of the International Workshop on Automatic Speaker Recognition, Identification, and Verification*, Martigny (Switzerland), April 5-7, 1994, pp. 189-192.

[5] J. L. McClelland and D. E. Rumelhart, "Explorations in Parallel Distributed Processing," *MIT Press*, 1988, pp. 132-135.

[6] T. Poston, C. Lee, Y. Choie, and Y. Kwon, "Local minima and Backpropagation," *Proc. of the IEEE-IJCNN91*, vol. II, Seattle, 8-12 July 1991, pp. 173-176.

[7] X. H. Yu, "Can Backpropagation Error Surface Not Have Local Minima?," *IEEE Trans. on Neural Network*, vol. 3, no. 6, November 1992, pp. 1019-1020.

Learning in Multilayered Networks Used as Autoassociators *

M. Bianchini, P. Frasconi, and M. Gori, *Member, IEEE*

Dipartimento di Sistemi e Informatica, Universitá di Firenze
Via di Santa Marta 3 - 50139 Firenze - Italy
Tel. +39 (55) 479.6265 - Fax +39 (55) 479.6363
e-mail : marco@ingfi1.ing.unifi.it

Abstract

Gradient descent learning algorithms may get stuck in local minima, thus making the learning sub-optimal. In this paper, we focus attention on multilayered networks used as autoassociators and show some relationships with classical linear autoassociators. In addition, using the theoretical framework of our previous research [3], we derive a condition which is met at the end of the learning process and show that this condition has a very intriguing geometrical meaning in the pattern space.

Index Terms- Autoassociators, Backpropagation, multilayered networks.

I Introduction

It is quite a common opinion that in order to use multilayered networks successfully, a thorough understanding of the learning process is needed. Among the theoretical problems related to *Backpropagation* (BP), that of the error surface local minima is one of the most important. It has been investigated by numerous researchers in the attempt of finding examples of local minima and conditions for guaranteeing local minima free error surfaces. A general analysis on that problem, which also contains numerous references, can be found in [3], where some conditions are given for guaranteeing that the error surfaces are local minima free in the case of *pyramidal networks*. Roughly speaking the BP optimal convergence is guaranteed for networks with *many inputs* and *many hidden units*. In [3], the already cited analysis is specialized for the case of linearly separable patterns, which is likely to hold for patterns represented by "many coordinates". In [6, 7], the absence of local minima is guaranteed for networks with one hidden layer and as many hidden units as patterns. A similar condition leads obviously to nets with "many hidden units". In this paper, we propose an analysis of the learning for multilayered networks that turns out to be particularly useful for autoassociators. We discuss the role of the hidden neuron nonlinearity by showing that, in spite of linear autoassociators, any network configuration autoassociates patterns of a limited region. Concerning the problem of local minima however, we give an example that shows the

*This research was partially supported by MURST 40%.

problems introduced by the neuron nonlinearity and that supports some experimental failures reported in literature (see e.g. [2, 5]). The complex shape of the cost surface motivates accurate analyses for gaining more information on the learning process. Unfortunately, in the case of autoassociators the general analysis proposed in [3] does not hold, since it is conceived specifically for pyramidal networks. As for the studies by Yu [7] and Poston [6], they are likely to require a number of hidden units greater than the number of inputs (outputs), which contradicts the basic principle of autoassociators to compress the input information at the hidden layer. The study of the learned configurations proposed in this paper is based on the theoretical framework suggested in [3]. We give a condition that is necessarily met at the end of the learning process (*end-of-learning condition*). It holds in any case, no matter what number of hidden units is chosen, and with no relationship to the network interpolation capabilities. It involves the *output coordinate pattern correlation* and the *input-output coordinate pattern correlation* [1]. It is shown that, as a particular case, this condition has a very intriguing geometrical meaning in terms of *projection operators*. In the special case of linear hidden units, these operators become projection matrices and the proposed condition establishes the global optimization of the learning process.

II Notations and vectorial formulation

In this section, we define the formalism adopted throughout the paper. Basically, three entities need to be defined: a network $\mathcal{N}$, a learning environment $\mathcal{L}_e$ (set of data used for learning), and a cost index E_T.

- *Network* $\mathcal{N}$.
  It has a multilayered architecture. With reference to index l, we distinguish among the input layer ($l = 0$), the output layer ($l = L$), and hidden layers ($0 < l < L$). The number of neurons per layer is denoted by $n(l)$. Each neuron of layer l is referred to by its index $i(l)$: $i(l) = 1, \ldots, n(l)$. When pattern "t" is presented at the input, for each neuron, we consider $a_{i(l)}(t)$ (neuron $i(l)$'s activation), and $x_{i(l)}(t)$ (neuron $i(l)$'s output). The activation is computed by propagating forward the outputs of the previous level neurons as follows [2]:

$$a_{i(l)}(t) = w_{i(l)} \; + \sum_{j(l-1)=1}^{n(l-1)} w_{i(l),j(l-1)} x_{j(l-1)}(t), \tag{1}$$

  where $w_{i(l),j(l-1)}$ denotes the weight of the link between the neurons $j(l-1), i(l)$ and $w_{i(l)}$ is the threshold. In many cases, when using the network as an autoassociator, the output of neuron $i(l)$ is related to the activation by

$$x_{i(l)} = f(a_{i(l)}) \text{ for } l < L \text{ and } x_{i(L)} = a_{i(L)}, \tag{2}$$

  where $f(\cdot) : R \rightarrow [\underline{d}, \overline{d}]$ is a C^2 "squashing-like" function with a positive first derivative, matching the asymptotical conditions $\lim_{a \rightarrow +\infty} f(a) = \overline{d}$ and $\lim_{a \rightarrow -\infty} f(a) = \underline{d}$.

  Sometimes it is assumed that the outputs are also processed with a squashing function, that is $x_{i(L)} = f(a_{i(L)})$. In that case, the inputs must be properly preprocessed so as to be constrained in $[\underline{d}, \overline{d}]$, in order to impose the supervision required by the autoassociators.

[1] See section IV for a formal definition of these terms.

[2] In the sequel, both layer and pattern indices may be omitted for the sake of simplicity.

- *Learning Environment* $\mathcal{L}_e$.
  We use a set of supervised data for learning. In general, it is convenient to think of it as a collection of T input/output pairs for network $\mathcal{N}$. Since the network is used as an autoassociator, the targets $\mathcal{D}(t)$ are equal to the inputs, and therefore, the learning environment can be defined solely in terms of the inputs:
  $$\mathcal{L}_e \doteq \left\{ X_0(t) \in R^{n(0)}, t = 1, \ldots, T \right\}.$$
  $X_0(t)$ is the input pattern, also used as target.

- *Cost index.*
  For a given $\mathcal{L}_e$, the input-output data fitting is measured by means of the cost function
  $$E_T \doteq \frac{1}{2} \sum_{t=1}^{T} E_t = \frac{1}{2} \sum_{t=1}^{T} \sum_{j=1}^{n(L)} [x_{j(0)}(t) - x_{j(L)}(t)]^2. \tag{3}$$

In order to understand the basic result proposed in this paper, some more symbols must be introduced which allow us to deal with a compact vectorial formulation of the problem.

1. $X_{i(l)} \doteq [x_{i(l)}(1), \ldots, x_{i(l)}(T)]' \in R^T$ stores the output of neuron $i(l)$ for all the T patterns of the learning environment.

2. These vectors can be collected in the matrix $\mathcal{X}_l \doteq [X_{1(l)} \cdots X_{n(l)} \ \Pi] \in R^{T, n(l)+1}$, $0 \leq l \leq L$, where $\Pi \doteq [1, \ldots, 1]' \in R^T$ is used to deal with biases.

3. $w_{i(l), j(l-1)}$ is the weight which connects neuron $j(l-1)$ to neuron $i(l)$. The associated matrix $\mathcal{W}_{l-1} \in R^{n(l), n(l-1)}$ is referred to as *layer weight matrix.*

4. The *delta-error* is defined by $y_{i(l)}(t) \doteq \partial E_t / \partial a_{i(l)}(t)$, $Y_{i(l)} \doteq [y_{i(l)}(1), \ldots, y_{i(l)}(T)]'$, and $\mathcal{Y}_l \doteq [Y_{1(l)} \cdots Y_{n(l)}] \in R^{T, n(l)}$.

5. We assume that there is no connection which jumps a layer. Therefore, for weights connecting layers l and $l-1$ the gradient can be represented by a matrix $\mathcal{G}_{l-1} \in R^{n(l-1)+1, n(l)}$, whose generic element $g(j(l-1), i(l))$ is given by $\partial E_T / \partial w_{i(l), j(l-1)}$ if $j(l-1) \leq n(l-1)$, and $\partial E_T / \partial w_{i(l)}$ if $j(l-1) = n(l-1) + 1$ (bias term gradient contribution).

With these definitions the gradient matrix can be computed as follows [3]:

$$\mathcal{G}_{l-1} = \mathcal{X}'_{l-1} \mathcal{Y}_l, \qquad l = 1, \ldots, L. \tag{4}$$

III Linear versus nonlinear autoassociator

In this section, we discuss the relationships between linear and nonlinear autoassociators in order to give some general suggestions concerning the application to actual problems.

- **Nonlinear autoassociators perform pattern autoassociation of "clustered" regions**
  This property does not hold for linear autoassociators and consequently they can not be used for modeling patterns clustered somewhere in the space, since they only autoassociate linear spaces. In order to establish this concept more formally, let us define the following *ϵ-autoassociation* concept.

Definition
Let $\varepsilon \in \mathcal{R}^+$, $X_0(t)$, $X_L(t) \in R^{n(0)}$ be, where $X_L(t) = \mathcal{H}(X_0(t))$ and $\mathcal{H}$ is the mapping provided by the autoassociator. We say that $X_0(t)$ is ε-autoassociated if [3]

$$\frac{\| X_L(t) - X_0(t) \|_\infty}{\| X_L(t) \|_\infty} < \varepsilon. \tag{5}$$

Theorem 1 *Let us consider autoassociators with linear output units (see equation (2)). If the pattern $X_0(t)$ is ε-autoassociated, then $\forall\ \lambda : \lambda \geq \lambda_\varepsilon = (1+\varepsilon)||\mathcal{W}_{L-1}||_\infty / ||X_0(t)||_\infty$ the pattern $\lambda X_0(t)$ is not ε-autoassociated.*

Proof
Let us consider the following inequality:

$$\frac{\| \mathcal{H}(\lambda X_0(t)) - \lambda X_0(t) \|_\infty}{\| \mathcal{H}(\lambda X_0(t)) \|_\infty} > \varepsilon. \tag{6}$$

In order to prove the theorem we must show that there exists $\lambda_\varepsilon \in \mathcal{R}^+$ such that the previous inequality holds $\forall\ \lambda \geq \lambda_\varepsilon$. Since $||\mathcal{H}(\lambda X_0(t)) - \lambda X_0(t)||_\infty = ||\lambda X_0(t) - \mathcal{H}(\lambda X_0(t))||_\infty \geq ||\lambda X_0(t)||_\infty - ||\mathcal{H}(\lambda X_0(t))||_\infty$, the search of λ satisfying inequality (6) can take place also in

$$\lambda ||X_0(t)||_\infty \geq (1+\varepsilon)||\mathcal{H}(\lambda X_0(t))||_\infty. \tag{7}$$

If we find λ satisfying (7) then inequality (6) is automatically verified. Since the network has linear outputs, it is simply $X_L(t) = \mathcal{W}_{L-1} X_{L-1}(t)$. As a result

$$||X_L(t)||_\infty \leq ||\mathcal{W}_{L-1}||_\infty ||X_{L-1}(t)||_\infty \leq ||\mathcal{W}_{L-1}||_\infty. \tag{8}$$

From inequalities (7) and (8) it follows that $\lambda X_0(t)$ is not ε-autoassociated provided that $\lambda \geq \lambda_\varepsilon = (1+\varepsilon)||\mathcal{W}_{L-1}||_\infty / ||X_0(t)||_\infty$. ■

The meaning of this theoretical result is that only limited regions in the pattern space are ε-autoassociated, that is the nonlinear autoassociator with linear outputs performs a sort of *clustering* in the pattern space. Notice that, in the case of linear autoassociators, the previous property does not hold, since if $X_0(t)$ is ε-autoassociated then $\lambda X_0(t)$ is ε-autoassociated no matter what λ we choose.

A practical implication of this result is that nonlinear autoassociators with linear outputs turns out to be suitable for applications in which we need performing a clustering in the pattern space. For example, successful results have recently been found in the field of speech verification [4], where an autoassociator was used for modeling each speaker voice. When trained on the associated speaker voice, the nonlinear autoassociators performed very well in rejecting "false" speakers.

- **The cost function associated with nonlinear autoassociators can be populated by local minima**
  Baldi and Hornik [1] have proven that linear autoassociators produce local minima free error surfaces. The experimental feeling of many researchers however, is that this property does not hold

[3] We have chosen $||\ ||_\infty$ for simplicity. It is worth mentioning that the result remains valid for any equivalent norm, simply introducing the *ad hoc* constant in eq. (8).

when introducing nonlinearity in the hidden layer (see e.g. [2, 5]). Unfortunately, as in the case of pyramidal networks [3], the nonlinearity of the hidden units introduces local minima in the cost function that can make the learning very hard (see the Appendix).

IV The end-of-learning condition

Our analysis focuses on stationary points. We investigate what happens when the gradient of E_T is "zero", in order to discover what configurations cause learning algorithms to get stuck. A first general result can be derived by inspecting solely the null gradient condition w.r.t. the weights connecting the last layer. This result can be established for a network of any number of layers, but is reported here only for the case of one hidden layer, typically used in practice.

Theorem 2 *Let $\mathcal{N}$ be a network used as a linear output autoassociator for the learning environment $\mathcal{L}_e$. Under this assumption, the following condition* [4]

$$\mathcal{X}_2'\mathcal{X}_2 = \mathcal{X}_2'\mathcal{X}_0 \tag{9}$$

holds at the end of the learning.

Proof

At the end of the learning the gradient w.r.t. all the weights connected to the output layer must be null, $\mathcal{X}_1'\mathcal{Y}_2 = 0$, which, in turn, implies

$$\mathcal{W}_1\mathcal{X}_1'\mathcal{Y}_2 = \mathcal{X}_2'\mathcal{Y}_2 = 0. \tag{10}$$

Finally, since the outputs are not "squashed" and because of the Backpropagation relationship applied at the output layer, $\mathcal{Y}_2 = \mathcal{X}_2 - \mathcal{X}_0$, the thesis of the theorem follows:

$$\mathcal{X}_2'(\mathcal{X}_2 - \mathcal{X}_0) = \mathcal{X}_2'\mathcal{X}_2 - \mathcal{X}_2'\mathcal{X}_0 = 0. \tag{11}$$

■

At the end of the learning process, this condition imposes the equality of the *output coordinate correlation* $\mathcal{X}_2'\mathcal{X}_2$ and the *input-output coordinate correlation* $\mathcal{X}_2'\mathcal{X}_0$. In this paper, equation (9) is referred to as *end-of-learning* condition. Some remarks are worth mentioning.

Remark 1: If the network is capable of interpolating perfectly the given learning environment then the solution $\mathcal{X}_2 = \mathcal{X}_0$ obviously exists for (9). Moreover, if the same pattern $\tilde{t}$ is presented at the input T times ($rank\mathcal{X}_0 = 1$) equation (9) yields $Tx_{i(2)}^2(\tilde{t}) = Tx_{i(0)}(\tilde{t})x_{i(2)}(\tilde{t})\ \forall\ i(0) = i(2) = 1, \ldots, n(0)$, that implies $x_{i(2)}(\tilde{t}) = x_{i(0)}(\tilde{t})\ \forall\ i$, thus guaranteeing optimal learning. (Configuration in which some $x_{i(2)}(\tilde{t}) = 0$ are saddle points [5]. In the case of patterns clustered in the neighborhood of $\tilde{t}$ one may expect no dramatic changes for the cost function, particularly for "small clusters").

[4] Notice that the same result holds for any feedforward network with target $\mathcal{D}(t) \neq \mathcal{X}_0(t)$. In that case $\mathcal{X}_2'\mathcal{X}_2 = \mathcal{X}_2'\mathcal{D}$ holds at the end of the learning process.

[5] As it is easily verifiable in the analogous case of a pattern set containing only one pattern. In this case, for the 2–1–2 autoassociator (see fig. 1 in the Appendix), calculating the second derivative along $n = (n_1, n_2, n_3, n_4)'$, $\frac{d^2E}{dn^2}$, as an n_1-polynomial and supposing, for example, $x_{i(2)} = 0\ \ \forall\ i(2)$, the discriminant is $\frac{\Delta}{4} = f'^2(n_3x_{11}^2 + n_4x_{11}x_{12})^2 \geq 0$, thus assuring that the derivative changes in sign. Moreover such a point may be a zero for equation (9), but not for the gradient.

Remark 2: In order to understand the meaning of condition (9), let us take the trace on both sides and exchange the sums w.r.t. the coordinate i and the pattern t:

$$\sum_{t=1}^{T} \| X_2(t) \|^2 = \sum_{t=1}^{T} X_2'(t) X_0(t). \tag{12}$$

This relationship has a very intriguing geometrical meaning which comes out by considering the network transformation $\mathcal{H} : R^n \rightarrow R^n : \mathcal{H}(X_0(t)) \rightarrow X_2(t)$:

$$\sum_{t=1}^{T} \| \mathcal{H}(X_0(t)) \|^2 = \sum_{t=1}^{T} X_0'(t) \mathcal{H}(X_0(t)). \tag{13}$$

In the case of linear hidden units operator $\mathcal{H}$ is linear and the associated matrix H satisfies the relationship $H^2 = H$. Hence H is a *projection matrix* and represents the optimal solution. By analogy with Linear Algebra, operator $\mathcal{H}$ will be referred to as a *projection operator*. Equation (12) can also be given a further interesting interpretation. If we exploit the identity $\| X_2(t) - X_0(t) \|^2 = \| X_2(t) \|^2 - 2X_0'(t)X_2(t) + \| X_0(t) \|^2$, equation (13) becomes:

$$\sum_{t=1}^{T} \| X_0(t) \|^2 = \sum_{t=1}^{T} \| \delta(t) \|^2 + \sum_{t=1}^{T} \| X_2(t) \|^2, \tag{14}$$

being $\delta(t) \doteq X_0(t) - X_2(t)$. This equation makes it clear that the "efficiency" of the learning depends on "energy" $\sum_{t=1}^{T} \| \delta(t) \|^2$. Since matrix $\mathcal{X}_2 \in R^{T, n(2)}$ has rank $n(1)$, at most, for an efficient learning, a dimensionality-reduction of the input pattern must be possible.

Remark 3: Equation (12) can be rewritten for fully nonlinear autoassociators (with nonlinearity even at the output layer). In this case, by imposing the null gradient condition $g_{i(2),j(1)} = 0$, the following condition follows

$$g_{i(2),j(1)} = \sum_t x_{j(1)} y_{i(2)} = \sum_t x_{j(1)} x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) = 0, \tag{15}$$

since for squashing function f, $f' = f(1-f)$ holds. If we right-multiply $w_{i(2),j(1)}$ and sum up w.r.t. $j(1)$ we obtain

$$\begin{aligned} \textstyle\sum_{j(1)} w_{i(2),j(1)} g_{i(2),j(1)} &= \textstyle\sum_t (\sum_{j(1)} w_{i(2),j(1)} x_{j(1)}) x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t a_{i(2)} x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t (\log \frac{x_{i(2)}}{1 - x_{i(2)}}) x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t \Phi(x_{i(2)})[x_{i(2)} - x_{i(0)}] = 0, \end{aligned} \tag{16}$$

being $\Phi(x) \doteq x(1-x) \log \frac{x}{1-x}$. Summing up over $i(2)$ yields

$$\sum_t \sum_{i(2)} \Phi(x_{i(2)})[x_{i(2)} - x_{i(0)}] = 0. \tag{17}$$

This equation is the natural generalization of the projection theorem stated by (12). Since $lim_{x \rightarrow 1} \Phi(x) = \Phi(1/2) = 0$, equation (17) suggests that, unlike what happens in the case of linear outputs, patterns with values close to 0.5 and 1 do not contribute to the end-of-learning condition.

Remark 4: It is worth mentioning that the *end-of-learning* relation was obtained only by exploiting the condition of null gradient on stationary points w.r.t. the output layer. Let us consider the implications of imposing the null gradient at the hidden layer. For nonlinear autoassociator, we cannot write the global null gradient condition, although for neuron $i(1)$ $\mathcal{X}_0'\mathcal{D}_{i(1)}(\mathcal{X}_2 - \mathcal{X}_0)W_{i(1)} = 0$ holds, where $\mathcal{D}_{i(1)} = diag(f'(a_{i(1)}))$. Finally, right-multiplying by $\mathcal{X}_1'$ yields

$$\widehat{\mathcal{X}}_{0_{i(1)}}'(\mathcal{X}_2 - \mathcal{X}_0)X_{2_{i(1)}}' = 0, \tag{18}$$

being $\widehat{\mathcal{X}}_{0_{i(1)}}'$ a *local-scaled* version of $\mathcal{X}_0'$. For linear autoassociators equation (18) becomes $(\mathcal{X}_2'\mathcal{X}_2 - \mathcal{X}_0'\mathcal{X}_0)\mathcal{X}_2' = 0$. Finally, in the *linear symmetrical* case, $(\mathcal{W} \doteq \mathcal{W}_0 = \mathcal{W}_1')$, the null gradient condition on output layer is $\mathcal{X}_1'\mathcal{Y}_2 = 0$, or equivalently $\mathcal{W}_0\mathcal{X}_0'\mathcal{Y}_2 = 0$, while, $\mathcal{G}_0 = 0$ yields $\mathcal{X}_0'\mathcal{Y}_1 = \mathcal{X}_0'\mathcal{Y}_2\mathcal{W}_1 = 0$. Then matrices $\mathcal{X}_0'\mathcal{Y}_2$ and $\mathcal{Y}_2'\mathcal{X}_0$ must belong to the kernel of matrix $\mathcal{W}$. Because of the BP relationship on the output layer $\mathcal{X}_0'\mathcal{Y}_2 = \mathcal{X}_0'(\mathcal{X}_2 - \mathcal{X}_0) = \mathcal{X}_0'\mathcal{W}'\mathcal{W}\mathcal{X}_0 - \mathcal{X}_0'\mathcal{X}_0$ is a symmetric matrix. As a consequence $\mathcal{G}_0 = \mathcal{X}_0'\mathcal{Y}_2\mathcal{W}_1 = 0$ implies $\mathcal{W}_1'\mathcal{Y}_2'\mathcal{X}_0 = \mathcal{W}_0\mathcal{X}_0'\mathcal{Y}_2 = \mathcal{G}_1 = 0$ and, therefore, in this case, equation (9) define completely all the stationary points.

V Conclusions

This paper discusses on the relationships between linear and nonlinear autoassociators. It turns out that the better computational capabilities of nonlinear autoassociators can be very useful in practice, but, unfortunately, their learning can be seriously plagued by local minima in the error surface.
This paper gives some insights for better understanding the configurations "magically" learned by algorithms like Backpropagation. A condition is given that must necessarily be met at the end of the learning, no matter what algorithm is used. This condition has a very intriguing geometrical meaning in the pattern space.

Acknowledgements
We would like to thank Marco Maggini for his very useful comments and suggestions.

APPENDIX

Suboptimal Learning in Nonlinear Autoassociators

Let us consider the network depicted in fig. 1 and suppose it is fed by

$$\mathcal{X}_0 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}. \tag{19}$$

As $rank\mathcal{X}_2$ is at most 1, suppose a possible solution to the autoassociation problem is

$$\mathcal{X}_2 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \tag{20}$$

where the first pattern is mismatched to the pair $(0,0)$, whereas the second one is exactly autoassociated. In the case of two patterns, cost function E can be written as

$$\begin{aligned} E = \quad \tfrac{1}{2} \quad & \{[f(x_{11}w_{31} + x_{12}w_{32})w_{43} - x_{11}]^2 + [f(x_{11}w_{31} + x_{12}w_{32})w_{53} - x_{12}]^2 + \\ & [f(x_{21}w_{31} + x_{22}w_{32})w_{43} - x_{21}]^2 + [f(x_{21}w_{31} + x_{22}w_{32})w_{53} - x_{22}]^2\}, \end{aligned} \tag{21}$$

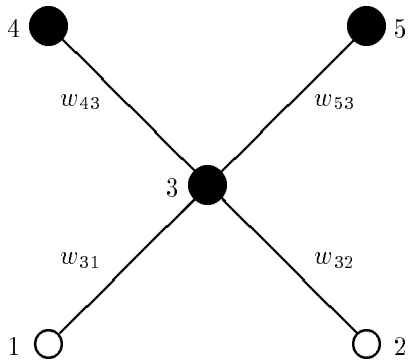

Figure 1: Autoassociator 2–1–2.

where function f is the identity in the case of linear autoassociator, otherwise $f = \frac{1}{1+e^{-x}}$. Thus, the associated gradient vector is:

$$\nabla E = \begin{bmatrix} \sum_{i=1}^{2} f'(a_{3i}) x_{i1} [(x_{3i} w_{43} - x_{i1}) w_{43} + (x_{3i} w_{53} - x_{i2}) w_{53}] \\ \sum_{i=1}^{2} f'(a_{3i}) x_{i2} [(x_{3i} w_{43} - x_{i1}) w_{43} + (x_{3i} w_{53} - x_{i2}) w_{53}] \\ \sum_{i=1}^{2} [x_{3i} w_{43} - x_{i1}] x_{3i} \\ \sum_{i=1}^{2} [x_{3i} w_{53} - x_{i2}] x_{3i} \end{bmatrix}, \tag{22}$$

being $a_{3i} = x_{i1} w_{31} + x_{i2} w_{32}$, $x_{3i} = f(a_{3i})$ and $f'(a_{3i}) = 1$ for linear autoassociator. As can be proven by substitution in equation (22), $\mathcal{X}_2$ is not a solution in the linear case. Forcing the null gradient condition yields $x_{31} = 0$ from the last two components and $w_{43} = -w_{53}$ from the first. Being $w_{43} \neq 0$, from $x_{32} w_{43} = 1$ $x_{32} = \frac{1}{w_{43}}$ follows. As a consequence, $x_{32} w_{53} = -\frac{1}{w_{43}} w_{43} = -1 = 0$, which is impossible.
In the nonlinear autoassociator the null gradient condition on the last two components produces again $x_{31} = 0$ [6], which, in turn, implies $f'(a_{31}) = 0$, $f''(a_{31}) = 0$, and so on. Moreover $x_{31} = 0$ also guarantees that the first two components of the gradient are null. As a consequence $\mathcal{X}_2$ is a stationary point. The Hessian matrix evaluated at this point

$$H = \begin{bmatrix} f'^2(a_{32})(w_{43}^2 + w_{53}^2) & 0 & f'(a_{32}) & 0 \\ 0 & 0 & 0 & 0 \\ f'(a_{32}) & 0 & x_{32}^2 & 0 \\ 0 & 0 & 0 & x_{32}^2 \end{bmatrix} \tag{23}$$

has null determinant. Nevertheless, writing the expression for the related second derivative along $n = (n_1, n_2, n_3, n_4)'$ yields

$$\frac{d^2 E}{dn^2} = n' H n = n_1^2 [f'^2(a_{32})(w_{43}^2 + w_{53}^2)] + 2 n_1 [n_3 f'(a_{32})] + [(n_3^2 + n_4^2) x_{32}^2], \tag{24}$$

which may be interpreted as a second degree polynomial on n_1 with constant positive sign, as $\frac{\Delta}{4} = -n_4^2 f'^2(a_{32})$. Therefore $\mathcal{X}_2$ is the output associated with a "*local minima configuration*", being $(0, 1, 0, 0)$, the direction corresponding to the infinite weight w_{32}, the only one for which $\frac{d^2 E}{dn^2}$ is null. (So there is a closed canyon with only one flat way out, which, numerically, acts as a minimum).

[6] In this case, $x_{31} = 0$ is simply accomplished choosing arbitrarily $w_{31} \neq \infty$ while $w_{32} \to -\infty$.

References

[1] P. Baldi and K. Hornik, "Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima," *Neural Networks*, vol. 2, 1989, pp. 53-58.

[2] H. Bourlard and Y. Kamp, "Auto-Association by Multilayer Perceptrons and Singular Value Decomposition," *Biological Cybernetics*, n. 59, 1988, pp. 291-294.

[3] M. Gori and A. Tesi, "On the problem of local minima in Backpropagation," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-14, no. 1, January 1992, pp. 76-86.

[4] L. Lastrucci, M. Gori, and G. Soda, "Neural Autoassociators for Phoneme-Based Speaker Verification," *Proc. of the International Workshop on Automatic Speaker Recognition, Identification, and Verification*, Martigny (Switzerland), April 5-7, 1994, pp. 189-192.

[5] J. L. McClelland and D. E. Rumelhart, "Explorations in Parallel Distributed Processing," *MIT Press*, 1988, pp. 132-135.

[6] T. Poston, C. Lee, Y. Choie, and Y. Kwon, "Local minima and Backpropagation," *Proc. of the IEEE-IJCNN91*, vol. II, Seattle, 8-12 July 1991, pp. 173-176.

[7] X. H. Yu, "Can Backpropagation Error Surface Not Have Local Minima?," *IEEE Trans. on Neural Network*, vol. 3, no. 6, November 1992, pp. 1019-1020.

Learning in Multilayered Networks Used as Autoassociators *

M. Bianchini, P. Frasconi, and M. Gori, *Member, IEEE*

Dipartimento di Sistemi e Informatica, Universitá di Firenze
Via di Santa Marta 3 - 50139 Firenze - Italy
Tel. +39 (55) 479.6265 - Fax +39 (55) 479.6363
e-mail : marco@ingfi1.ing.unifi.it

Abstract

Gradient descent learning algorithms may get stuck in local minima, thus making the learning sub-optimal. In this paper, we focus attention on multilayered networks used as autoassociators and show some relationships with classical linear autoassociators. In addition, using the theoretical framework of our previous research [3], we derive a condition which is met at the end of the learning process and show that this condition has a very intriguing geometrical meaning in the pattern space.

Index Terms- Autoassociators, Backpropagation, multilayered networks.

I Introduction

It is quite a common opinion that in order to use multilayered networks successfully, a thorough understanding of the learning process is needed. Among the theoretical problems related to *Backpropagation* (BP), that of the error surface local minima is one of the most important. It has been investigated by numerous researchers in the attempt of finding examples of local minima and conditions for guaranteeing local minima free error surfaces. A general analysis on that problem, which also contains numerous references, can be found in [3], where some conditions are given for guaranteeing that the error surfaces are local minima free in the case of *pyramidal networks*. Roughly speaking the BP optimal convergence is guaranteed for networks with *many inputs* and *many hidden units*. In [3], the already cited analysis is specialized for the case of linearly separable patterns, which is likely to hold for patterns represented by "many coordinates". In [6, 7], the absence of local minima is guaranteed for networks with one hidden layer and as many hidden units as patterns. A similar condition leads obviously to nets with "many hidden units". In this paper, we propose an analysis of the learning for multilayered networks that turns out to be particularly useful for autoassociators. We discuss the role of the hidden neuron nonlinearity by showing that, in spite of linear autoassociators, any network configuration autoassociates patterns of a limited region. Concerning the problem of local minima however, we give an example that shows the

*This research was partially supported by MURST 40%.

problems introduced by the neuron nonlinearity and that supports some experimental failures reported in literature (see e.g. [2, 5]). The complex shape of the cost surface motivates accurate analyses for gaining more information on the learning process. Unfortunately, in the case of autoassociators the general analysis proposed in [3] does not hold, since it is conceived specifically for pyramidal networks. As for the studies by Yu [7] and Poston [6], they are likely to require a number of hidden units greater than the number of inputs (outputs), which contradicts the basic principle of autoassociators to compress the input information at the hidden layer. The study of the learned configurations proposed in this paper is based on the theoretical framework suggested in [3]. We give a condition that is necessarily met at the end of the learning process (*end-of-learning condition*). It holds in any case, no matter what number of hidden units is chosen, and with no relationship to the network interpolation capabilities. It involves the *output coordinate pattern correlation* and the *input-output coordinate pattern correlation* [1]. It is shown that, as a particular case, this condition has a very intriguing geometrical meaning in terms of *projection operators*. In the special case of linear hidden units, these operators become projection matrices and the proposed condition establishes the global optimization of the learning process.

II Notations and vectorial formulation

In this section, we define the formalism adopted throughout the paper. Basically, three entities need to be defined: a network $\mathcal{N}$, a learning environment $\mathcal{L}_e$ (set of data used for learning), and a cost index E_T.

- *Network* $\mathcal{N}$.
  It has a multilayered architecture. With reference to index l, we distinguish among the input layer ($l = 0$), the output layer ($l = L$), and hidden layers ($0 < l < L$). The number of neurons per layer is denoted by $n(l)$. Each neuron of layer l is referred to by its index $i(l)$: $i(l) = 1, \ldots, n(l)$. When pattern "t" is presented at the input, for each neuron, we consider $a_{i(l)}(t)$ (neuron $i(l)$'s activation), and $x_{i(l)}(t)$ (neuron $i(l)$'s output). The activation is computed by propagating forward the outputs of the previous level neurons as follows [2]:

$$a_{i(l)}(t) = w_{i(l)} \; + \sum_{j(l-1)=1}^{n(l-1)} w_{i(l),j(l-1)} x_{j(l-1)}(t), \tag{1}$$

  where $w_{i(l),j(l-1)}$ denotes the weight of the link between the neurons $j(l-1), i(l)$ and $w_{i(l)}$ is the threshold. In many cases, when using the network as an autoassociator, the output of neuron $i(l)$ is related to the activation by

$$x_{i(l)} = f(a_{i(l)}) \text{ for } l < L \text{ and } x_{i(L)} = a_{i(L)}, \tag{2}$$

  where $f(\cdot) : R \rightarrow [\underline{d}, \overline{d}]$ is a C^2 "squashing-like" function with a positive first derivative, matching the asymptotical conditions $\lim_{a \rightarrow +\infty} f(a) = \overline{d}$ and $\lim_{a \rightarrow -\infty} f(a) = \underline{d}$.

  Sometimes it is assumed that the outputs are also processed with a squashing function, that is $x_{i(L)} = f(a_{i(L)})$. In that case, the inputs must be properly preprocessed so as to be constrained in $[\underline{d}, \overline{d}]$, in order to impose the supervision required by the autoassociators.

[1] See section IV for a formal definition of these terms.

[2] In the sequel, both layer and pattern indices may be omitted for the sake of simplicity.

- *Learning Environment* $\mathcal{L}_e$.
  We use a set of supervised data for learning. In general, it is convenient to think of it as a collection of T input/output pairs for network $\mathcal{N}$. Since the network is used as an autoassociator, the targets $\mathcal{D}(t)$ are equal to the inputs, and therefore, the learning environment can be defined solely in terms of the inputs:
  $$\mathcal{L}_e \doteq \left\{ X_0(t) \in R^{n(0)}, t = 1, \ldots, T \right\}.$$
  $X_0(t)$ is the input pattern, also used as target.

- *Cost index.*
  For a given $\mathcal{L}_e$, the input-output data fitting is measured by means of the cost function
  $$E_T \doteq \frac{1}{2} \sum_{t=1}^{T} E_t = \frac{1}{2} \sum_{t=1}^{T} \sum_{j=1}^{n(L)} [x_{j(0)}(t) - x_{j(L)}(t)]^2. \tag{3}$$

In order to understand the basic result proposed in this paper, some more symbols must be introduced which allow us to deal with a compact vectorial formulation of the problem.

1. $X_{i(l)} \doteq [x_{i(l)}(1), \ldots, x_{i(l)}(T)]' \in R^T$ stores the output of neuron $i(l)$ for all the T patterns of the learning environment.

2. These vectors can be collected in the matrix $\mathcal{X}_l \doteq [X_{1(l)} \cdots X_{n(l)} \ \Pi] \in R^{T, n(l)+1}$, $0 \leq l \leq L$, where $\Pi \doteq [1, \ldots, 1]' \in R^T$ is used to deal with biases.

3. $w_{i(l), j(l-1)}$ is the weight which connects neuron $j(l-1)$ to neuron $i(l)$. The associated matrix $\mathcal{W}_{l-1} \in R^{n(l), n(l-1)}$ is referred to as *layer weight matrix.*

4. The *delta-error* is defined by $y_{i(l)}(t) \doteq \partial E_t / \partial a_{i(l)}(t)$, $Y_{i(l)} \doteq [y_{i(l)}(1), \ldots, y_{i(l)}(T)]'$, and $\mathcal{Y}_l \doteq [Y_{1(l)} \cdots Y_{n(l)}] \in R^{T, n(l)}$.

5. We assume that there is no connection which jumps a layer. Therefore, for weights connecting layers l and $l-1$ the gradient can be represented by a matrix $\mathcal{G}_{l-1} \in R^{n(l-1)+1, n(l)}$, whose generic element $g(j(l-1), i(l))$ is given by $\partial E_T / \partial w_{i(l), j(l-1)}$ if $j(l-1) \leq n(l-1)$, and $\partial E_T / \partial w_{i(l)}$ if $j(l-1) = n(l-1) + 1$ (bias term gradient contribution).

With these definitions the gradient matrix can be computed as follows [3]:

$$\mathcal{G}_{l-1} = \mathcal{X}'_{l-1} \mathcal{Y}_l, \qquad l = 1, \ldots, L. \tag{4}$$

III Linear versus nonlinear autoassociator

In this section, we discuss the relationships between linear and nonlinear autoassociators in order to give some general suggestions concerning the application to actual problems.

- **Nonlinear autoassociators perform pattern autoassociation of "clustered" regions**
  This property does not hold for linear autoassociators and consequently they can not be used for modeling patterns clustered somewhere in the space, since they only autoassociate linear spaces. In order to establish this concept more formally, let us define the following *ϵ-autoassociation* concept.

Definition
Let $\varepsilon \in \mathcal{R}^+$, $X_0(t)$, $X_L(t) \in R^{n(0)}$ be, where $X_L(t) = \mathcal{H}(X_0(t))$ and $\mathcal{H}$ is the mapping provided by the autoassociator. We say that $X_0(t)$ is ε-autoassociated if [3]

$$\frac{\| X_L(t) - X_0(t) \|_\infty}{\| X_L(t) \|_\infty} < \varepsilon. \tag{5}$$

Theorem 1 *Let us consider autoassociators with linear output units (see equation (2)). If the pattern $X_0(t)$ is ε-autoassociated, then $\forall\ \lambda : \lambda \geq \lambda_\varepsilon = (1+\varepsilon)||\mathcal{W}_{L-1}||_\infty / ||X_0(t)||_\infty$ the pattern $\lambda X_0(t)$ is not ε-autoassociated.*

Proof
Let us consider the following inequality:

$$\frac{\| \mathcal{H}(\lambda X_0(t)) - \lambda X_0(t) \|_\infty}{\| \mathcal{H}(\lambda X_0(t)) \|_\infty} > \varepsilon. \tag{6}$$

In order to prove the theorem we must show that there exists $\lambda_\varepsilon \in \mathcal{R}^+$ such that the previous inequality holds $\forall\ \lambda \geq \lambda_\varepsilon$. Since $||\mathcal{H}(\lambda X_0(t)) - \lambda X_0(t)||_\infty = ||\lambda X_0(t) - \mathcal{H}(\lambda X_0(t))||_\infty \geq ||\lambda X_0(t)||_\infty - ||\mathcal{H}(\lambda X_0(t))||_\infty$, the search of λ satisfying inequality (6) can take place also in

$$\lambda ||X_0(t)||_\infty \geq (1+\varepsilon)||\mathcal{H}(\lambda X_0(t))||_\infty. \tag{7}$$

If we find λ satisfying (7) then inequality (6) is automatically verified. Since the network has linear outputs, it is simply $X_L(t) = \mathcal{W}_{L-1} X_{L-1}(t)$. As a result

$$||X_L(t)||_\infty \leq ||\mathcal{W}_{L-1}||_\infty ||X_{L-1}(t)||_\infty \leq ||\mathcal{W}_{L-1}||_\infty. \tag{8}$$

From inequalities (7) and (8) it follows that $\lambda X_0(t)$ is not ε-autoassociated provided that $\lambda \geq \lambda_\varepsilon = (1+\varepsilon)||\mathcal{W}_{L-1}||_\infty / ||X_0(t)||_\infty$. ■

The meaning of this theoretical result is that only limited regions in the pattern space are ε-autoassociated, that is the nonlinear autoassociator with linear outputs performs a sort of *clustering* in the pattern space. Notice that, in the case of linear autoassociators, the previous property does not hold, since if $X_0(t)$ is ε-autoassociated then $\lambda X_0(t)$ is ε-autoassociated no matter what λ we choose.

A practical implication of this result is that nonlinear autoassociators with linear outputs turns out to be suitable for applications in which we need performing a clustering in the pattern space. For example, successful results have recently been found in the field of speech verification [4], where an autoassociator was used for modeling each speaker voice. When trained on the associated speaker voice, the nonlinear autoassociators performed very well in rejecting "false" speakers.

- **The cost function associated with nonlinear autoassociators can be populated by local minima**
  Baldi and Hornik [1] have proven that linear autoassociators produce local minima free error surfaces. The experimental feeling of many researchers however, is that this property does not hold

[3] We have chosen $||\ ||_\infty$ for simplicity. It is worth mentioning that the result remains valid for any equivalent norm, simply introducing the *ad hoc* constant in eq. (8).

when introducing nonlinearity in the hidden layer (see e.g. [2, 5]). Unfortunately, as in the case of pyramidal networks [3], the nonlinearity of the hidden units introduces local minima in the cost function that can make the learning very hard (see the Appendix).

IV The end-of-learning condition

Our analysis focuses on stationary points. We investigate what happens when the gradient of E_T is "zero", in order to discover what configurations cause learning algorithms to get stuck. A first general result can be derived by inspecting solely the null gradient condition w.r.t. the weights connecting the last layer. This result can be established for a network of any number of layers, but is reported here only for the case of one hidden layer, typically used in practice.

Theorem 2 *Let $\mathcal{N}$ be a network used as a linear output autoassociator for the learning environment $\mathcal{L}_e$. Under this assumption, the following condition* [4]

$$\mathcal{X}_2'\mathcal{X}_2 = \mathcal{X}_2'\mathcal{X}_0 \tag{9}$$

holds at the end of the learning.

Proof

At the end of the learning the gradient w.r.t. all the weights connected to the output layer must be null, $\mathcal{X}_1'\mathcal{Y}_2 = 0$, which, in turn, implies

$$\mathcal{W}_1\mathcal{X}_1'\mathcal{Y}_2 = \mathcal{X}_2'\mathcal{Y}_2 = 0. \tag{10}$$

Finally, since the outputs are not "squashed" and because of the Backpropagation relationship applied at the output layer, $\mathcal{Y}_2 = \mathcal{X}_2 - \mathcal{X}_0$, the thesis of the theorem follows:

$$\mathcal{X}_2'(\mathcal{X}_2 - \mathcal{X}_0) = \mathcal{X}_2'\mathcal{X}_2 - \mathcal{X}_2'\mathcal{X}_0 = 0. \tag{11}$$

■

At the end of the learning process, this condition imposes the equality of the *output coordinate correlation* $\mathcal{X}_2'\mathcal{X}_2$ and the *input-output coordinate correlation* $\mathcal{X}_2'\mathcal{X}_0$. In this paper, equation (9) is referred to as *end-of-learning* condition. Some remarks are worth mentioning.

Remark 1: If the network is capable of interpolating perfectly the given learning environment then the solution $\mathcal{X}_2 = \mathcal{X}_0$ obviously exists for (9). Moreover, if the same pattern $\tilde{t}$ is presented at the input T times ($rank\mathcal{X}_0 = 1$) equation (9) yields $Tx_{i(2)}^2(\tilde{t}) = Tx_{i(0)}(\tilde{t})x_{i(2)}(\tilde{t})\ \forall\ i(0) = i(2) = 1, \ldots, n(0)$, that implies $x_{i(2)}(\tilde{t}) = x_{i(0)}(\tilde{t})\ \forall\ i$, thus guaranteeing optimal learning. (Configuration in which some $x_{i(2)}(\tilde{t}) = 0$ are saddle points [5]. In the case of patterns clustered in the neighborhood of $\tilde{t}$ one may expect no dramatic changes for the cost function, particularly for "small clusters").

[4] Notice that the same result holds for any feedforward network with target $\mathcal{D}(t) \neq \mathcal{X}_0(t)$. In that case $\mathcal{X}_2'\mathcal{X}_2 = \mathcal{X}_2'\mathcal{D}$ holds at the end of the learning process.

[5] As it is easily verifiable in the analogous case of a pattern set containing only one pattern. In this case, for the 2–1–2 autoassociator (see fig. 1 in the Appendix), calculating the second derivative along $n = (n_1, n_2, n_3, n_4)'$, $\frac{d^2E}{dn^2}$, as an n_1-polynomial and supposing, for example, $x_{i(2)} = 0\ \ \forall\ i(2)$, the discriminant is $\frac{\Delta}{4} = f'^2(n_3x_{11}^2 + n_4x_{11}x_{12})^2 \geq 0$, thus assuring that the derivative changes in sign. Moreover such a point may be a zero for equation (9), but not for the gradient.

Remark 2: In order to understand the meaning of condition (9), let us take the trace on both sides and exchange the sums w.r.t. the coordinate i and the pattern t:

$$\sum_{t=1}^{T} \| X_2(t) \|^2 = \sum_{t=1}^{T} X_2'(t) X_0(t). \tag{12}$$

This relationship has a very intriguing geometrical meaning which comes out by considering the network transformation $\mathcal{H} : R^n \rightarrow R^n : \mathcal{H}(X_0(t)) \rightarrow X_2(t)$:

$$\sum_{t=1}^{T} \| \mathcal{H}(X_0(t)) \|^2 = \sum_{t=1}^{T} X_0'(t) \mathcal{H}(X_0(t)). \tag{13}$$

In the case of linear hidden units operator $\mathcal{H}$ is linear and the associated matrix H satisfies the relationship $H^2 = H$. Hence H is a *projection matrix* and represents the optimal solution. By analogy with Linear Algebra, operator $\mathcal{H}$ will be referred to as a *projection operator*. Equation (12) can also be given a further interesting interpretation. If we exploit the identity $\| X_2(t) - X_0(t) \|^2 = \| X_2(t) \|^2 - 2X_0'(t)X_2(t) + \| X_0(t) \|^2$, equation (13) becomes:

$$\sum_{t=1}^{T} \| X_0(t) \|^2 = \sum_{t=1}^{T} \| \delta(t) \|^2 + \sum_{t=1}^{T} \| X_2(t) \|^2, \tag{14}$$

being $\delta(t) \doteq X_0(t) - X_2(t)$. This equation makes it clear that the "efficiency" of the learning depends on "energy" $\sum_{t=1}^{T} \| \delta(t) \|^2$. Since matrix $\mathcal{X}_2 \in R^{T, n(2)}$ has rank $n(1)$, at most, for an efficient learning, a dimensionality-reduction of the input pattern must be possible.

Remark 3: Equation (12) can be rewritten for fully nonlinear autoassociators (with nonlinearity even at the output layer). In this case, by imposing the null gradient condition $g_{i(2),j(1)} = 0$, the following condition follows

$$g_{i(2),j(1)} = \sum_t x_{j(1)} y_{i(2)} = \sum_t x_{j(1)} x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) = 0, \tag{15}$$

since for squashing function f, $f' = f(1-f)$ holds. If we right-multiply $w_{i(2),j(1)}$ and sum up w.r.t. $j(1)$ we obtain

$$\begin{aligned} \textstyle\sum_{j(1)} w_{i(2),j(1)} g_{i(2),j(1)} &= \textstyle\sum_t (\sum_{j(1)} w_{i(2),j(1)} x_{j(1)}) x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t a_{i(2)} x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t (\log \frac{x_{i(2)}}{1 - x_{i(2)}}) x_{i(2)} (1 - x_{i(2)})(x_{i(2)} - x_{i(0)}) \\ &= \textstyle\sum_t \Phi(x_{i(2)})[x_{i(2)} - x_{i(0)}] = 0, \end{aligned} \tag{16}$$

being $\Phi(x) \doteq x(1-x) \log \frac{x}{1-x}$. Summing up over $i(2)$ yields

$$\sum_t \sum_{i(2)} \Phi(x_{i(2)})[x_{i(2)} - x_{i(0)}] = 0. \tag{17}$$

This equation is the natural generalization of the projection theorem stated by (12). Since $lim_{x \rightarrow 1} \Phi(x) = \Phi(1/2) = 0$, equation (17) suggests that, unlike what happens in the case of linear outputs, patterns with values close to 0.5 and 1 do not contribute to the end-of-learning condition.

Remark 4: It is worth mentioning that the *end-of-learning* relation was obtained only by exploiting the condition of null gradient on stationary points w.r.t. the output layer. Let us consider the implications of imposing the null gradient at the hidden layer. For nonlinear autoassociator, we cannot write the global null gradient condition, although for neuron $i(1)$ $\mathcal{X}_0'\mathcal{D}_{i(1)}(\mathcal{X}_2 - \mathcal{X}_0)W_{i(1)} = 0$ holds, where $\mathcal{D}_{i(1)} = diag(f'(a_{i(1)}))$. Finally, right-multiplying by $\mathcal{X}_1'$ yields

$$\widehat{\mathcal{X}}_{0_{i(1)}}'(\mathcal{X}_2 - \mathcal{X}_0)X_{2_{i(1)}}' = 0, \tag{18}$$

being $\widehat{\mathcal{X}}_{0_{i(1)}}'$ a *local-scaled* version of $\mathcal{X}_0'$. For linear autoassociators equation (18) becomes $(\mathcal{X}_2'\mathcal{X}_2 - \mathcal{X}_0'\mathcal{X}_0)\mathcal{X}_2' = 0$. Finally, in the *linear symmetrical* case, $(\mathcal{W} \doteq \mathcal{W}_0 = \mathcal{W}_1')$, the null gradient condition on output layer is $\mathcal{X}_1'\mathcal{Y}_2 = 0$, or equivalently $\mathcal{W}_0\mathcal{X}_0'\mathcal{Y}_2 = 0$, while, $\mathcal{G}_0 = 0$ yields $\mathcal{X}_0'\mathcal{Y}_1 = \mathcal{X}_0'\mathcal{Y}_2\mathcal{W}_1 = 0$. Then matrices $\mathcal{X}_0'\mathcal{Y}_2$ and $\mathcal{Y}_2'\mathcal{X}_0$ must belong to the kernel of matrix $\mathcal{W}$. Because of the BP relationship on the output layer $\mathcal{X}_0'\mathcal{Y}_2 = \mathcal{X}_0'(\mathcal{X}_2 - \mathcal{X}_0) = \mathcal{X}_0'\mathcal{W}'\mathcal{W}\mathcal{X}_0 - \mathcal{X}_0'\mathcal{X}_0$ is a symmetric matrix. As a consequence $\mathcal{G}_0 = \mathcal{X}_0'\mathcal{Y}_2\mathcal{W}_1 = 0$ implies $\mathcal{W}_1'\mathcal{Y}_2'\mathcal{X}_0 = \mathcal{W}_0\mathcal{X}_0'\mathcal{Y}_2 = \mathcal{G}_1 = 0$ and, therefore, in this case, equation (9) define completely all the stationary points.

V Conclusions

This paper discusses on the relationships between linear and nonlinear autoassociators. It turns out that the better computational capabilities of nonlinear autoassociators can be very useful in practice, but, unfortunately, their learning can be seriously plagued by local minima in the error surface.
This paper gives some insights for better understanding the configurations "magically" learned by algorithms like Backpropagation. A condition is given that must necessarily be met at the end of the learning, no matter what algorithm is used. This condition has a very intriguing geometrical meaning in the pattern space.

Acknowledgements
We would like to thank Marco Maggini for his very useful comments and suggestions.

APPENDIX

Suboptimal Learning in Nonlinear Autoassociators

Let us consider the network depicted in fig. 1 and suppose it is fed by

$$\mathcal{X}_0 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}. \tag{19}$$

As $rank\mathcal{X}_2$ is at most 1, suppose a possible solution to the autoassociation problem is

$$\mathcal{X}_2 = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \tag{20}$$

where the first pattern is mismatched to the pair $(0,0)$, whereas the second one is exactly autoassociated. In the case of two patterns, cost function E can be written as

$$\begin{aligned} E = \quad \tfrac{1}{2} \quad & \{[f(x_{11}w_{31} + x_{12}w_{32})w_{43} - x_{11}]^2 + [f(x_{11}w_{31} + x_{12}w_{32})w_{53} - x_{12}]^2 + \\ & [f(x_{21}w_{31} + x_{22}w_{32})w_{43} - x_{21}]^2 + [f(x_{21}w_{31} + x_{22}w_{32})w_{53} - x_{22}]^2\}, \end{aligned} \tag{21}$$

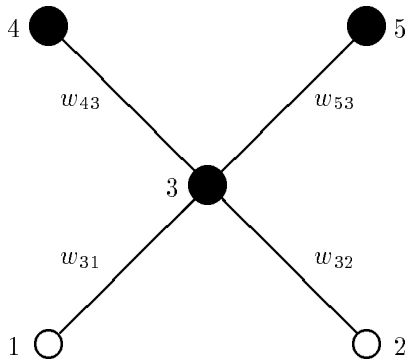

Figure 1: Autoassociator 2–1–2.

where function f is the identity in the case of linear autoassociator, otherwise $f = \frac{1}{1+e^{-x}}$. Thus, the associated gradient vector is:

$$\nabla E = \begin{bmatrix} \sum_{i=1}^{2} f'(a_{3i}) x_{i1} [(x_{3i} w_{43} - x_{i1}) w_{43} + (x_{3i} w_{53} - x_{i2}) w_{53}] \\ \sum_{i=1}^{2} f'(a_{3i}) x_{i2} [(x_{3i} w_{43} - x_{i1}) w_{43} + (x_{3i} w_{53} - x_{i2}) w_{53}] \\ \sum_{i=1}^{2} [x_{3i} w_{43} - x_{i1}] x_{3i} \\ \sum_{i=1}^{2} [x_{3i} w_{53} - x_{i2}] x_{3i} \end{bmatrix}, \tag{22}$$

being $a_{3i} = x_{i1} w_{31} + x_{i2} w_{32}$, $x_{3i} = f(a_{3i})$ and $f'(a_{3i}) = 1$ for linear autoassociator. As can be proven by substitution in equation (22), $\mathcal{X}_2$ is not a solution in the linear case. Forcing the null gradient condition yields $x_{31} = 0$ from the last two components and $w_{43} = -w_{53}$ from the first. Being $w_{43} \neq 0$, from $x_{32} w_{43} = 1$ $x_{32} = \frac{1}{w_{43}}$ follows. As a consequence, $x_{32} w_{53} = -\frac{1}{w_{43}} w_{43} = -1 = 0$, which is impossible.
In the nonlinear autoassociator the null gradient condition on the last two components produces again $x_{31} = 0$ [6], which, in turn, implies $f'(a_{31}) = 0$, $f''(a_{31}) = 0$, and so on. Moreover $x_{31} = 0$ also guarantees that the first two components of the gradient are null. As a consequence $\mathcal{X}_2$ is a stationary point. The Hessian matrix evaluated at this point

$$H = \begin{bmatrix} f'^2(a_{32})(w_{43}^2 + w_{53}^2) & 0 & f'(a_{32}) & 0 \\ 0 & 0 & 0 & 0 \\ f'(a_{32}) & 0 & x_{32}^2 & 0 \\ 0 & 0 & 0 & x_{32}^2 \end{bmatrix} \tag{23}$$

has null determinant. Nevertheless, writing the expression for the related second derivative along $n = (n_1, n_2, n_3, n_4)'$ yields

$$\frac{d^2 E}{dn^2} = n' H n = n_1^2 [f'^2(a_{32})(w_{43}^2 + w_{53}^2)] + 2 n_1 [n_3 f'(a_{32})] + [(n_3^2 + n_4^2) x_{32}^2], \tag{24}$$

which may be interpreted as a second degree polynomial on n_1 with constant positive sign, as $\frac{\Delta}{4} = -n_4^2 f'^2(a_{32})$. Therefore $\mathcal{X}_2$ is the output associated with a "*local minima configuration*", being $(0, 1, 0, 0)$, the direction corresponding to the infinite weight w_{32}, the only one for which $\frac{d^2 E}{dn^2}$ is null. (So there is a closed canyon with only one flat way out, which, numerically, acts as a minimum).

[6] In this case, $x_{31} = 0$ is simply accomplished choosing arbitrarily $w_{31} \neq \infty$ while $w_{32} \to -\infty$.

References

[1] P. Baldi and K. Hornik, "Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima," *Neural Networks*, vol. 2, 1989, pp. 53-58.

[2] H. Bourlard and Y. Kamp, "Auto-Association by Multilayer Perceptrons and Singular Value Decomposition," *Biological Cybernetics*, n. 59, 1988, pp. 291-294.

[3] M. Gori and A. Tesi, "On the problem of local minima in Backpropagation," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. PAMI-14, no. 1, January 1992, pp. 76-86.

[4] L. Lastrucci, M. Gori, and G. Soda, "Neural Autoassociators for Phoneme-Based Speaker Verification," *Proc. of the International Workshop on Automatic Speaker Recognition, Identification, and Verification*, Martigny (Switzerland), April 5-7, 1994, pp. 189-192.

[5] J. L. McClelland and D. E. Rumelhart, "Explorations in Parallel Distributed Processing," *MIT Press*, 1988, pp. 132-135.

[6] T. Poston, C. Lee, Y. Choie, and Y. Kwon, "Local minima and Backpropagation," *Proc. of the IEEE-IJCNN91*, vol. II, Seattle, 8-12 July 1991, pp. 173-176.

[7] X. H. Yu, "Can Backpropagation Error Surface Not Have Local Minima?," *IEEE Trans. on Neural Network*, vol. 3, no. 6, November 1992, pp. 1019-1020.