Categorization and effective perceptron learning in feed-forward neural networks

J. Phys. A: Math. Gen. **33** (2000) 5809–5818. Printed in the UK PII: S0305-4470(00)15067-X

Categorization and effective perceptron learning in feed-forward neural networks

Henri Waelbroeck and Federico Zertuche

† Instituto de Ciencias Nucleares, UNAM Circuito Exterior, C U A, Postal 70-543, México DF 04510, México

‡ Instituto de Matemáticas, UNAM Unidad Cuernavaca, A.P. 273-3, Admon. 3 62251 Cuernavaca, Morelos, México

E-mail: hwael@nuclecu.unam.mx and zertuche@matcuer.unam.mx

Received 28 June 2000

Abstract. In the initial stages of back-propagation learning, weights in the first synaptic layer are small and hidden neurons operate in the linear regime; so the input–output map is then the same as that of a perceptron. As the weights increase, the hidden neurons begin to saturate; each saturation pattern represents a particular category of input vectors, for which the neural network (NN) locally behaves as a perceptron. The effective perceptron weights for each category are functions of the synaptic weights of the NN. We define an internal temperature as the inverse of the norm of the first synaptic matrix, and an entropy function that measures the disorder of the hidden-neuron layer. The learning curves are recast in the language of a condensation process. The entropy drops abruptly when the hidden neurons begin to saturate, marking a clear transition from the global perceptron state to an ordered state characterized by different local effective perceptrons in different categories. The problem of optimizing the effective perceptrons in different categories is frustrated.

1. Introduction

Since the successful development and implementation of replica symmetry techniques in recurrent neural networks (NN) for calculating a partition function [1], and also the implementation of central limit theorems [2], researchers have extended the thermodynamical formalism to the perceptron [3] and further to multilayer feed-forward neural networks (FNN) [4–6]. Interesting features, such as the plateau-like curves which one observes in the standard FNN learning by back-propagation, have been predicted [5].

The back-propagation algorithm for FNN has been widely and successfully used since its development [7]. Even though it is a gradient algorithm that can be fooled by local minima, its convergence is very quick and it is easy to program. Recently, the number of patterns that can be stored per weight have been calculated by statistical mechanics techniques for the fully connected committee machine [6]. However, until now there is no generally accepted prescription that one can follow in order to choose an architecture or avoid problems such as over-learning or getting trapped in local minima. Statistical mechanics techniques are very powerful and allow us to analyse the problems in a broad way, even though its range of validity is restricted to the thermodynamical limit, when the number of neurons in the input layer $N_1 \to \infty$. In this paper we study the problem of learning from a numerical perspective, and try to understand the different stages of learning and the type of input–output map that the NN produces in each stage.

Our aim in this paper is to study a three-layer FNN with N_1 in the input layer, N_2 in the hidden layer and 1 in the output layer, and focus our attention on the state of saturation of the hidden neurons, considering three different regimes for their activity: saturation to 0, saturation to 1, and a linear regime for small values of the activation potential. A key point is that there are classes of input vectors that leave the hidden neurons in a given state of saturation. These classes can be labelled by symbolic arrays giving a symbol in $\{0, 1, *\}$ for each hidden neuron: for example, in a FNN with four hidden neurons $(0, 0, 1, *)$ is the category of input vectors that saturate the first two hidden neurons to zero, the third to one, and leaves the fourth in a linear regime. The different input vectors in such a category might be very different (for example, from the point of view of the Euclidean norm in input space), but they are treated in a similar way by the neural network: the same hidden neurons saturate to 0, to 1, or not at all. Hidden neurons that saturate to 0 or 1 do not propagate small fluctuations in the inputs: they are effectively deactivated and can be replaced by a constant value. On the other hand, neurons that are operating in the linear regime will propagate the signal forward with a weight equal to the product of weights from the first and second synaptic layers. This leads to a picture of the FNN as a set of perceptrons, one for each category of inputs.

The categorization of input vectors by means of the hidden neuron activation patterns depends on the first synaptic matrix and therefore it evolves as one proceeds along the learning curve. The number of categories occupied with data vectors grows during the learning process. At advanced stages the number of categories becomes large and it is no longer possible to think of the learning process as separate processes taking place in each category, where the effective perceptron weights in that category get optimized independently. Instead, the task of optimizing effective perceptrons in the different categories becomes frustrated: improving the input–output map for one category of inputs reduces its performance for another category. This frustration effect has important consequences: for example, local minima would not exist without a frustrated learning problem.

In this paper we will make these arguments more concrete, and through this, offer an inside view of how the FNN operates at different stages of its learning curve, and how the back-propagation algorithm pushes the FNN along from one stage to the next.

In the following section we describe the architecture of the NN and describe the back-propagation algorithm, in section 3 the data set used for the training of the NN is presented. In section 4 we present the process of categorization by saturation of the hidden neurons and in section 5 we address the conclusions.

2. The back-propagation algorithm

We will consider a three-layer FNN with sigmoidal transfer functions, with N_1 inputs and N_2 hidden neurons, and no thresholds [7]:

$$y(x^\alpha) = g\left(\sum_{j=1}^{N_2} W_j^{(2)} x_{j(\alpha)}^{(2)}\right) \tag{2.1}$$

$$x_{j(\alpha)}^{(2)} = g\left(\sum_{i=1}^{N_1} W_{ji}^{(1)} x_i^\alpha\right) \tag{2.2}$$

$$g(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{2.3}$$

where x_i^α and $x_{j(\alpha)}^{(2)}$ are the αth input vectors on the ith input neuron and jth hidden neuron, respectively ($\alpha = 1, \ldots, P$ with P being the number of training vectors), and $y(x^\alpha)$ is the neural output. It is customary to insert a factor β in the argument of the exponential in the transfer function (2.3); however, this factor can be absorbed in the synaptic weights $W_{ji}^{(1)}$ and $W_j^{(2)}$.

The back-propagation algorithm is a steepest-descent minimization algorithm on the quadratic difference between the neural outputs $y(x^\alpha)$ and the known correct outputs y^α:

$$\epsilon = \frac{1}{P}\sum_{\alpha=1}^{P}(y(x^\alpha) - y^\alpha)^2. \tag{2.4}$$

Calculating the gradient of this error function with respect to the synaptic weights leads to the usual iterative algorithm for back-propagation, η being the learning rate:

$$W_j^{(2)} \to W_j^{(2)} - \eta(y(x^\alpha) - y^\alpha)y(x^\alpha)(1 - y(x^\alpha))x_{j(\alpha)}^{(2)} \tag{2.5}$$

$$W_{ji}^{(1)} \to W_{ji}^{(1)} - \eta(y(x^\alpha) - y^\alpha)y(x^\alpha)(1 - y(x^\alpha))W_j^{(2)}x_{j(\alpha)}^{(2)}(1 - x_{j(\alpha)}^{(2)})x_i^\alpha. \tag{2.6}$$

The training vectors (α) are presented in random order, one at a time and updating the synaptic weights at each step.

3. The dataset used

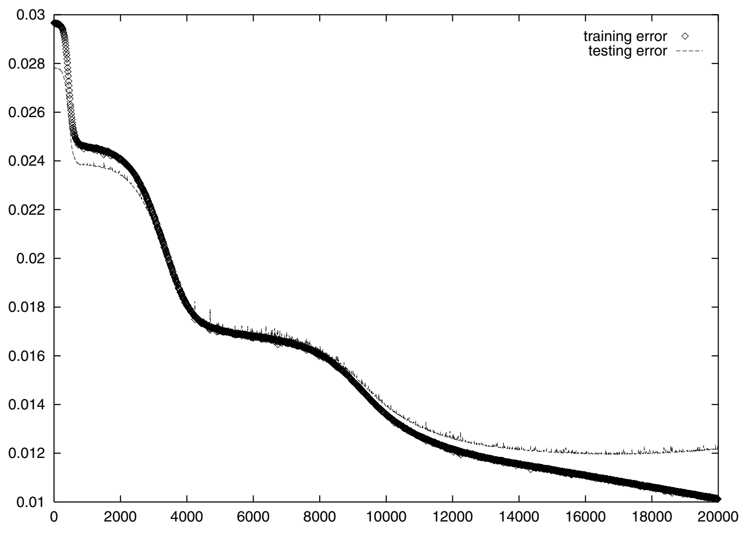
In order to address issues that are relevant in real-life applications of NN, we will use a time series forecasting problem that is sufficiently complex that one does not expect to achieve a very close fit to data, and yet is completely deterministic. A model that fits this picture is that which Lorenz introduced in 1991 to consider the applicability of phenomenological approaches to estimating the attractor dimension for systems which consist of a large number of weakly coupled chaotic subsystems: the model consists of seven weakly coupled copies of the classical Lorenz equations, i.e. 21 first-order differential equations for 21 variables [8]. We integrated this system using a fourth-order Runge–Kutta method, using the first minimum of mutual information for the first variable in Lorenz's system to set the sampling time and running the integrator until we have 4000 consecutive snapshots of this first variable.

4. Cooling curve

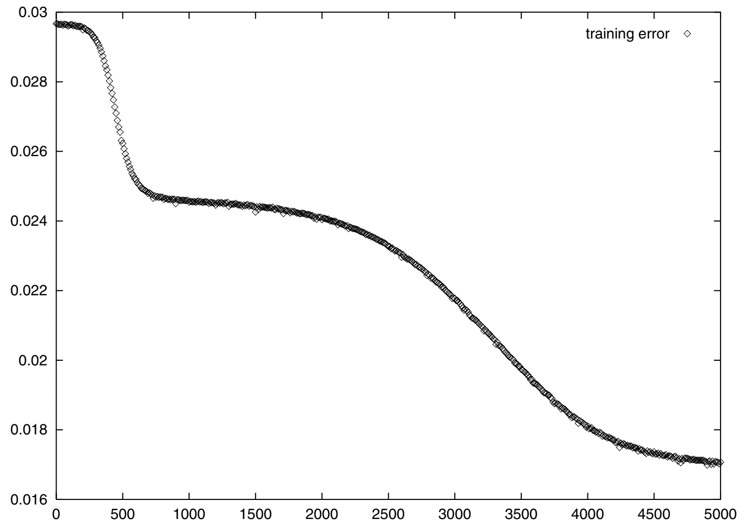
We will use a particular run to describe the whole process of learning: however, it is completely representative of the situation under study. With $\eta = 0.01$, we ran a back-propagation algorithm on a $N_1 = 24$, $N_2 = 16$ neural network with a small training data set of 976 training vectors, each one consisting of 24 consecutive values of the first variable in Lorenz's system. The remaining 3000 data vectors were saved for testing. The initial synaptic matrix was randomly generated with a uniform probability in the range $[-0.05, 0.05]$ for the second layer, and $[-0.0001, 0.0001]$ for the first synaptic layer. The learning curve on training and testing data are reproduced in figure 1(*a*): having taken a small training dataset, the over-learning phenomenon is particularly acute. Our purpose here, however, is to analyse the evolution of the FNN from left to right as it proceeds to learn the data.

The transfer function (2.3) of the hidden neurons can be approximated by a piecewise-linear function $g_0(x)$, the best approximation in terms of absolute error

$$\int_{-\infty}^{+\infty} |g(x) - g_0(x)|\,\mathrm{d}x \tag{4.1}$$



(*a*)



(*b*)

Figure 1. (*a*) The number of iterations of the back-propagation algorithm versus the error (2.4) is plotted for the training data and the testing data for over 20 000 iterations. (*b*) The number of iterations of the back-propagation algorithm versus the error (2.4) for the first 5000 iterations.

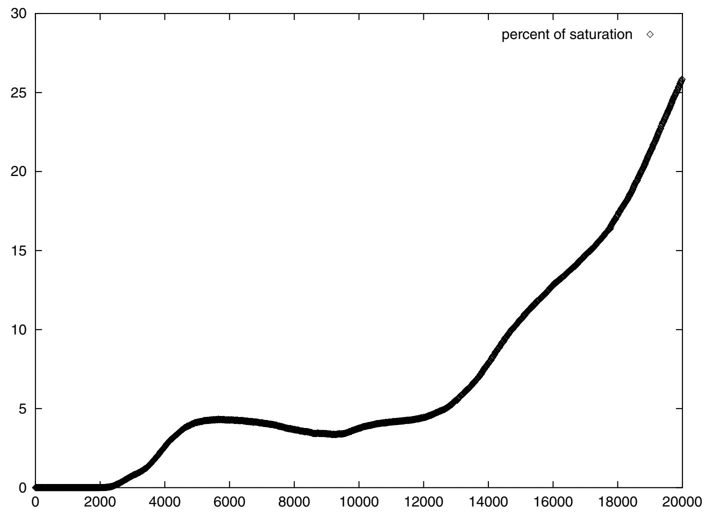
being the following:

$$g_0(x) = 0 \qquad -\infty < x \leqslant -\tfrac{5}{2} \tag{4.2}$$

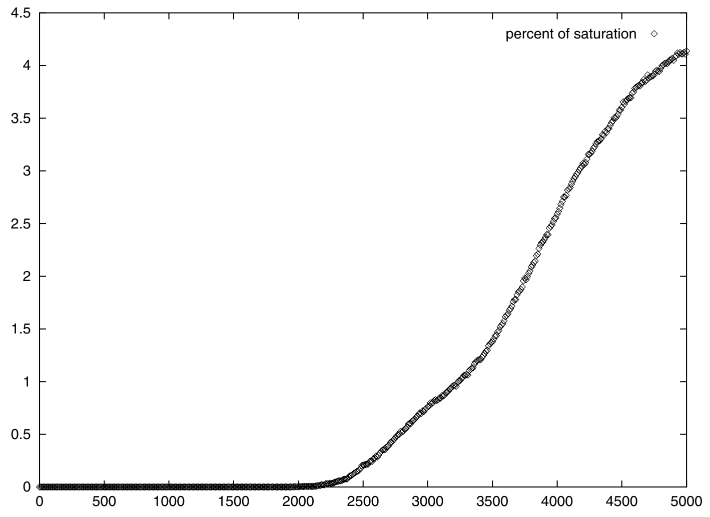
$$g_0(x) = \tfrac{1}{2} + \tfrac{1}{5}x \qquad -\tfrac{5}{2} < x < \tfrac{5}{2} \tag{4.3}$$

$$g_0(x) = 1 \qquad \tfrac{5}{2} \leqslant x < \infty. \tag{4.4}$$

With this approximation, one clearly distinguishes three different operating modes of the hidden neurons: saturation to zero, a quasi-linear map, and saturation to one. At first, of course, the synaptic weights being very small, the hidden neurons almost never reach saturation: as



(*a*)



(*b*)

Figure 2. (*a*) The number of iterations versus the per cent of saturation of the neurons in the hidden layer. (*b*) The number of iterations versus the per cent of saturation for the first 5000 iterations.

the synaptic weights increase following back-propagation, the degree of saturation increases and the FNN begins to discover nonlinear maps. In figure 2(*a*) the per cent of saturation of the neurons is plotted.

We will define the entropy of the hidden layer by analogy to spin systems: assuming that $x^{(2)}_{j(\alpha)}$ represents the mean value of a binary neuron with instantaneous activation values 0 or 1, for the αth input vector, the statistical entropy of the hidden layer is

$$S = \sum_{\alpha=1}^{P} \sum_{j=1}^{N_2} \{x^{(2)}_{j(\alpha)} \log(x^{(2)}_{j(\alpha)}) + (1 - x^{(2)}_{j(\alpha)}) \log(1 - x^{(2)}_{j(\alpha)})\}. \qquad (4.5)$$

We will also define the network's internal temperature as the inverse of the 1-norm of the

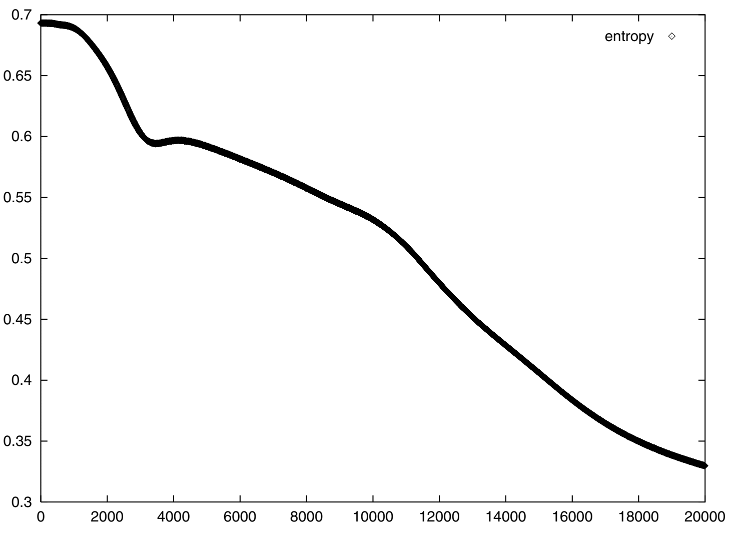


Figure 3. The number of iterations versus the entropy (4.5).

matrix $W^{(1)}$:

$$\beta = \|W^{(1)}\|_1 = \mathrm{Sup}_x \frac{\|Wx\|_1}{\|x\|_1}$$

where $\|x\|_1 = \sum_{i=1}^{N_1} |x_i|$ is the Manhattan norm of x (this choice of norm is natural here but similar results follow if one chooses other norms, such as the Euclidean norm, for example). This matrix norm can be computed as [9]

$$\|W^{(1)}\|_1 = \mathrm{Max}_{i=1,\ldots,N_1} \sum_{j=1}^{N_2} |W_{ji}^{(1)}|. \tag{4.6}$$

The entropy and inverse temperature are represented in figures 3 and 4, and in figure 5 the entropy is represented as a function of β.

As the process of learning goes on, one observes a series of plateaux in the learning curve, figure 1(a), on which the error remains almost constant. The same phenomenon occurs for β which controls the categorization process. This behaviour has already been observed by several authors in the context of statistical mechanics for FNN in on-line learning [5].

The first drop in the prediction error on the training set occurs at iteration number 200, as the network discovers the best quasi-linear fit to the data; at this stage there is no significant reduction in entropy as the hidden neurons remain mostly nonsaturated. Up to this point the FNN can be effectively replaced by a simple perceptron model. If we consider that each hidden neuron is approximated by the linear transfer function (4.3), then

$$y(x^\alpha) \approx \theta + \sum_{i=1}^{N_1} W_i^{ef} x_i^\alpha \tag{4.7}$$

where

$$W_i^{ef} = \tfrac{1}{25} \sum_{j=1}^{N_2} W_j^{(2)} W_{ji}^{(1)} \qquad \theta = \tfrac{1}{2} + \tfrac{1}{10} \sum_{j=1}^{N_2} W_j^{(2)}.$$

From 600–1960 iterations the prediction error remains almost constant. Since the number of effective parameters ($N_1 + 1$) is less than the number of synaptic weights, it is clear that

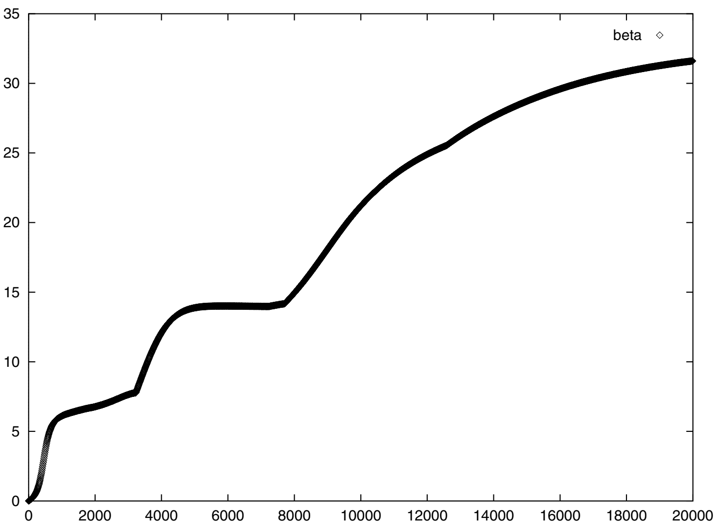


Figure 4. The number of iterations versus the inverse of the temperature β, equation (4.6).

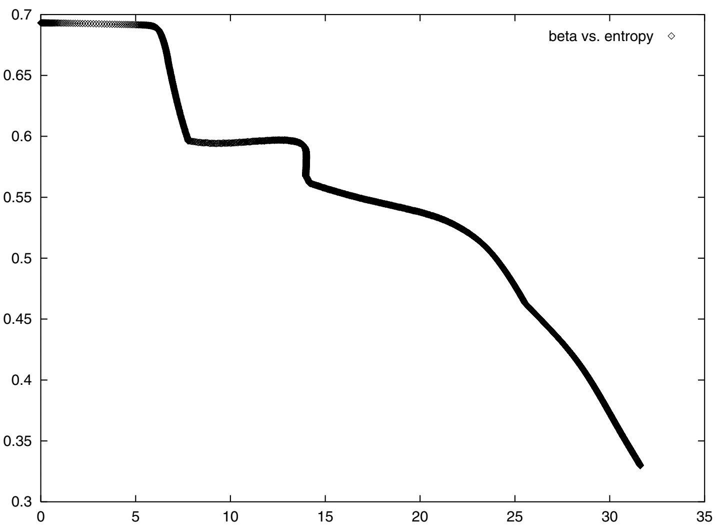


Figure 5. β versus the entropy.

there are many irrelevant directions in the error landscape at this point. One can view the learning process as a random drift on the horizontal subspace. As the random walk proceeds, the norm of the synaptic matrix increases as $\sqrt{N}$. Indeed, a log–log regression shows that $\beta \sim N^a$ with $a = 0.509 \pm 0.053$ and a correlation of 96.5%. Eventually, at 1960 iterations, the hidden neurons leave the linear regime (see figure 2(*b*)). At this point the network ceases to be equivalent to a single effective perceptron, the flat subspace disappears and the back-propagation algorithm can find negative gradient directions. This takes the network into a transition to a new optimum where some hidden neurons saturate. This is shown in figure 1(*b*) as a new decrease of the prediction error from 1960–4000 iterations, and in figure 2(*b*) as a rise of the per cent of saturation from 0–4. The transition from the global perceptron regime to saturated neurons can be observed in figure 5 as an abrupt drop in entropy with relatively little change in temperature at $\beta \approx 7$.

In the saturating regime the saturation patterns of the hidden neurons provides categorizing information; each category is defined by a symbolic word $(r_1, r_2, \ldots, r_{N_2})$ with $r_i \in \{0, \star, 1\}$ where 0 will denote neurons that are switched off, 1 denotes those that are switched on, and $\star$ denotes those neurons that are operating in the linear regime. For each hidden neuron we can distinguish two hyperplanes which separate regions of input space that saturate this neuron to 0 or 1, with the equations for the hyperplanes:

$$\sum_{i=1}^{N_1} W_{ji}^{(1)} x_i^{\alpha} = \pm\tfrac{5}{2}.$$

For all input vectors that produce the same hidden neuron activation pattern (or belong to the same *category*), the hidden neuron transfer functions can be replaced by the piecewise linear approximation, leading once again to an effective perceptron, but with the difference that, in this case, the perceptron we find will only approximate the FNN accurately for those input vectors in the given category. If we denote by $\{\star\}$ the hidden neurons that operate in the linear regime and $\{1\}$ those that saturate to one, the effective map in a given saturation category is a (local) effective perceptron (4.7) with parameters

$$W_i^{ef} = \tfrac{1}{25} \sum_{j\in\{\star\}} W_j^{(2)} W_{ji}^{(1)} \tag{4.8}$$

$$\theta = \tfrac{1}{2} + \tfrac{1}{5} \sum_{j\in\{1\}} W_j^{(2)} + \tfrac{1}{10} \sum_{j\in\{\star\}} W_j^{(2)}. \tag{4.9}$$

In this sense, one can view the transition from the quasi-linear to the nonlinear regime of the neural network as a transition from a global effective perceptron to local effective perceptrons, one in each hidden neuron saturation category.

From 4000–8000 iterations the prediction error remains once again almost constant, corresponding to a search in the nonlinear regime to the best local effective perceptrons. The temperature also stay almost constant while the entropy shows a new drop, corresponding to more order in the hidden layer of the NN as seen in figure 5. After iteration 8000 the NN uses this new training to decrease the temperature, the entropy and the prediction error over the training set, while the per cent of saturation increases. However the over-training also starts at this point (see figure 1(*a*)), indicating that further refinement of the phase space into smaller categories is not relevant.

This refinement of the categorization into a greater number of smaller and smaller categories comes as a consequence of the growth in the norm of the first synaptic matrix, and the corresponding increase in the frequency for saturating hidden neurons. The first refinements are clearly beneficial in the case of nonlinear problems, because they allow one to replace a global perceptron by several local perceptrons adjusted to different regions of the input vector space. The point, however, is that what drives the trend towards finer and finer partitions of the input space is more the random drift along neutral directions of the landscape than a definite learning gradient; this implies that FNNs continue to increase the hidden neurons saturation well beyond the point where the optimal input–output map is achieved, and this is the root of the over-fitting problem.

5. Conclusion

By introducing the concept of categorization in NN by means of the saturation of the neurons in the hidden layer, we have provided a tool to look inside the black box of the three-layer FNNs to understand how they work. The saturation concept of the hidden neurons is introduced by substituting the nonlinear transfer function (2.3) with a piecewise linear function (4.2)–(4.4), which is the best approximation in terms of the absolute error (4.1), and saying that a hidden neuron is saturated if its activation value is in the region of the linear functions (4.2) and (4.4) while it is nonsaturated if its activation value is in the linear region (4.3). For each category it is then possible to express the behaviour of the FNN in terms of an effective perceptron (4.7) with parameters (4.8) and (4.9).

Categorization permits us to study the learning process from the point of view of a cooling process, which shows abrupt changes in entropy at near constant temperature when the NN begins to perform categorization in resemblance of the phase transitions in thermodynamical systems away from equilibrium and in agreement with previous results found for on-line learning [5]. There are two main drops in entropy with almost constant temperature: one when the NN changes from a global perceptron and categorization starts, and the other when the NN is looking for the best local effective perceptron.

The picture of local effective perceptrons can be compared to the Farmer–Sidorowich phase space reconstruction method, which assumes local linearity in the input-to-output map [10]. It should be an improvement over this latter approach in the case when the first deviation from linearity is well modelled by the sigmoidal squashing function. Following this stream of thought, a four-layer hidden neuron can similarly be replaced by a set of effective three-layer FNNs, one in each category of input vectors. Of course, each one of these three-layer FNNs in turn will divide its input space in sub-categories. This shows how the addition of new layers to FNNs is related to a renormalization group type process where the space of input vectors gets partitioned into cells which then get partitioned into finer cells, etc. Of course, the frustration problem implies that the effective perceptrons in each cell are not mutually independent and therefore their effective weights cannot be optimized simultaneously.

Acknowledgments

This work was partially supported through the Spinoza Grant of the NWO (The Netherlands) and DGAPA-UNAM grant number IN105197. The authors wish to express their gratitude to Chris Stephens for numerous enlightening discussions, and to V Domínguez, E Sacristán and F Toledo for computational advice. The research effort was carried out in the context of a joint program involving Adaptive Technologies, NNCP (NN, Chaos and Prediction, http://www.nuclecu.unam.mx/ nncp) and the SIECyT (Sistema Incubador de Empresas con base Científica y Tecnológica). Adaptive Technologies develops and markets successful commercial applications of front line complex systems research in the area of computational finance (http://www.adaptrader.com).

References

[1] Amit D, Gutfreund H and Sompolinsky H 1985 *Phys. Rev. Lett.* **55** 1530
Amit D, Gutfreund H and Sompolinsky H 1987 *Ann. Phys., NY* **173** 30
Gardner E J 1987 *J. Phys. A: Math. Gen.* **20** 3453

[2] Geszti T 1990 *Physical Models of Neural Networks* (Singapore: World Scientific)
Zertuche F, López-Peña R and Waelbroeck H 1994 *J. Phys. A: Math. Gen.* **27** 1575–83
Zertuche F, López-Peña R and Waelbroeck H 1994 *J. Phys. A: Math. Gen.* **27** 5879–87

[3] Krogh A and Hertz J 1992 *J. Phys. A: Math. Gen.* **25** 1135–47
Horner H 1992 *Z. Phys.* B **87** 371–6
[4] Ahr M, Biehl M and Urbanczik R 1999 *Eur. Phys. J.* **10** 583–8
Schwarze H 1993 *J. Phys. A: Math. Gen.* **26** 5781–94
[5] Saad D and Solla S 1995 *Phys. Rev. Lett.* **74** 4337–40
Saad D and Solla S 1995 *Phys. Rev.* E **52** 4225–43
Biehl M, Riegler P and Wöhler C 1996 *J. Phys. A: Math. Gen.* **29** 4769–80
Riegler P and Biehl M 1995 *J. Phys. A: Math. Gen.* **28** L507
[6] Urbanczik R 1997 *J. Phys. A: Math. Gen.* **30** L387
[7] Hertz J, Krogh A and Palmer R G 1991 *Introduction to the Theory of Neural Computation* (Reading, MA: Addison-Wesley) and references therein
[8] Lorenz E N 1991 Dimension of weather and climate attractors *Nature* **353** 241 and references therein
[9] Atkinson K E 1988 *An Introduction to Numerical Analysis* (New York: Wiley) chapter 7
[10] Farmer J D and Sidorowich J J 1987 Predicting chaotic time series *Phys. Rev. Lett.* **59** 845–8