ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 211-216

Visualizing the Learning Process for Neural Networks

Raúl Rojas

Freie Universität Berlin
Fachbereich Mathematik und Informatik
Takustr. 9, Berlin 14195

Abstract. In this paper we present some visualization techniques which assist in understanding the iteration process of learning algorithms for neural networks. In the case of perceptron learning, we show that the algorithm can be visualized as a search on the surface of what we call the *boolean sphere*. In the case of backpropagation, we show that the iteration path is not just random noise, but that under certain circumstances it exhibits an interesting structure. Examples of on-line and off-line backpropagation iteration paths show that they are fractals.

1. Introduction

Neural networks have gained wide acceptance as an alternative computational paradigm in the last years. Many different applications have emerged and different kinds of learning algorithms have been proposed. But when first exposed to the basic tenants of this new computational model, scientists and students used to the traditional direct algorithmic methods of computer science have problems understanding just how the learning methods for neural networks manage to achieve convergence. If a package of ready-to-use subroutines is used to train the networks, the practitioner can only stare at the error curve and hope that it will eventually come down so that the whole learning process succeeds.
Over the course of the last years we have developed some visualization techniques which help to achieve an intuitive understanding of the way some learning algorithms work. This intuitive visualization is important, not only because it shows just what is happening, but also because it gives some clues about possible modifications of the learning algorithms [4]. In this paper we give three examples of these visualization techniques. We first show that the perceptron learning algorithm can be understood as a search on the surface of what we call the "boolean sphere". The visual approach makes the proof that the algorithm converges almost trivial. Our second example is on-line backpropagation: random selection of the training patterns defines a dynamical system whose path in weight space is effectively a fractal. Our third example is backpropagation with momentum. In this case the path in weight space is not just random noise but exhibits a convoluted structure. We provide some computer generated graphics of our visualizations.

2. The boolean sphere

In the case of a single perceptron with a step activation function, each input pattern and its target output define a hyperplane in weight space. To one side of the hyperplane lie all solutions in weight space for the pattern which has to be learned. The other side of the hyperplane can be ignored. Since several patterns define several hyperplanes, the solution region is the intersection of the semispaces in which solutions for each pattern are contained. The solution region is then convex and has the form of a polytope, like that shown in Fig. 1.

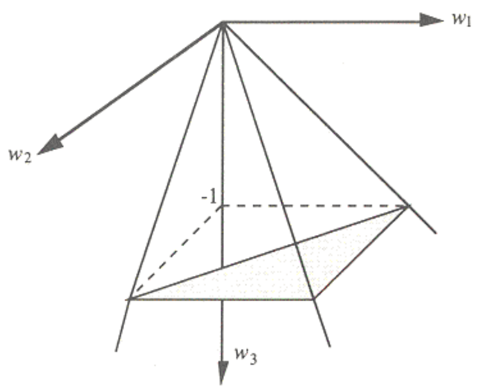

Fig. 1. Solution region in weight space

It is instructive to ask how many possible solution regions can be formed in the case of a perceptron with two inputs and three parameters (two weights and the threshold value). If the inputs can assume binary values, there are four possible combinations. This means that the three-dimensional weight space is cut by four planes associated with each one of the input patterns. Normalizing the weight vector, we can visualize the solution regions as regions on the surface of a sphere, the *boolean sphere*. It is now easy to see that the maximum number of regions is only fourteen. That means that two of the sixteen possible boolean functions cannot be computed by this perceptron. These are, of course, XOR and its negation. This provides a topological explanation of the uncomputability of these functions with a single perceptron. The question of deciding how many boolean functions of n arguments can be implemented with a single perceptron reduces then to a simple geometrical calculation.

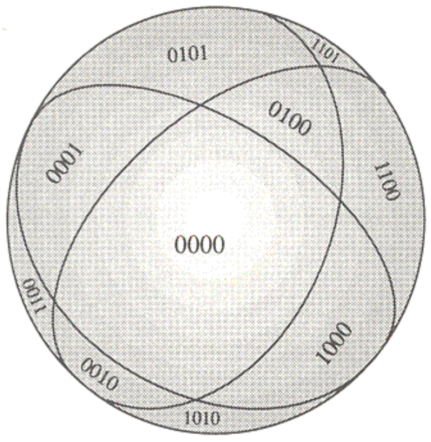

Fig. 2. The boolean sphere

ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 211-216

Figure 3 shows the back and front of the boolean sphere. The labels of the regions show which boolean functions can be calculated by taking a combination of weights lying in each region (the label 0000, for example, is associated with the boolean function which produces 0 for each of the four possible inputs, 1000 is the AND function, etc.). We can see that it is possible to go from one region to the other, each time correcting one bit of the output.

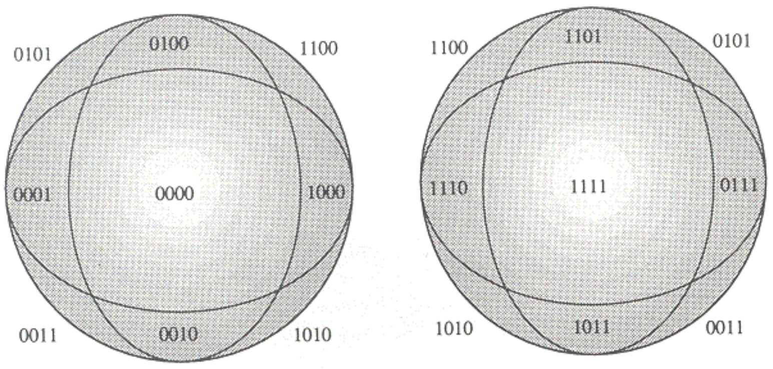

Fig. 3. Front and back views of the boolean sphere

3. Projections of the boolean sphere

It is even more interesting to look at the two-dimensional structure of the surface of the boolean sphere. This structure is represented in Fig. 4, a kind of topological stereometric projection of the boolean sphere. Note, again, that four circles can maximally define fourteen different solution regions in weight space.

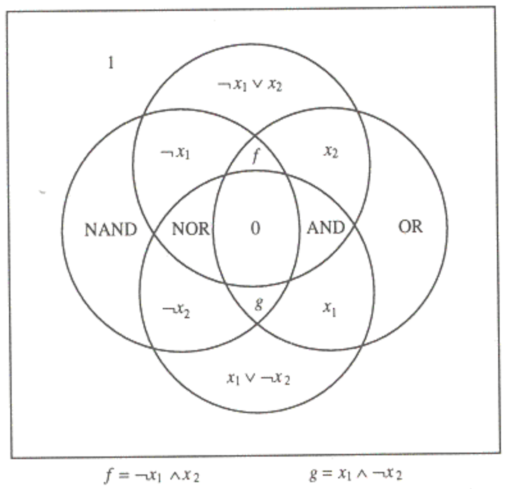

Fig. 4. Projection of the solution regions on two dimensions

Figure 5 shows the error function for a perceptron which must be trained to produce the function $(x_1, x_2) \to 1$. We can see that the error function has a single region which is a global maximum and a single region which is a global minimum. This is indeed a graphical proof that the perceptron learn algorithm converges, since we can always rotate the boolean sphere to get the same kind of projection. This visualization technique is superior to the one presented in [2].

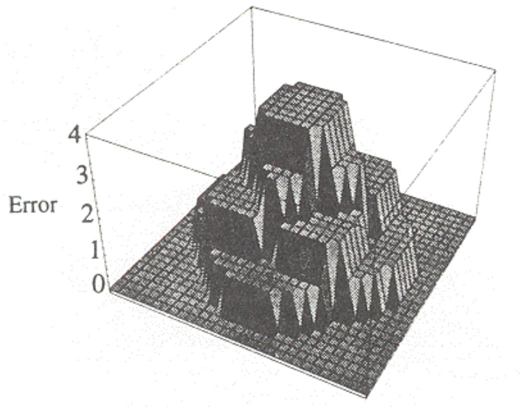

Fig. 5. Error surface for the function $(x_1, x_2) \rightarrow 1$

One optimization that immediately comes to mind is trying to make the relative sizes of the solution regions as equal as possible, since we do not know in advance which function should be learned. It can be shown that bipolar vectors produce the most symmetrical cuts of the boolean sphere. This common optimization for neural networks has thus this simple geometrical explanation. Table 1 shows the relative sizes of the solution regions for the first eight boolean functions in the case of a binary or a bipolar coding. The data was obtained using a Monte Carlo method. The same kind of calculation can be done for higher dimensional boolean spheres with similar results. The higher dimensionality of weight space increases the size differences between regions and thus the variance of the expected learning time.

Table 1: Relative sizes of the regions on the boolean sphere

	boolean function number							
coding	0	1	2	3	4	5	6	7
binary	26.83	2.13	4.18	4.13	4.17	4.22	0	4.13
bipolar	8.33	6.29	6.26	8.32	6.24	8.36	0	6.22

4. On-line Backpropagation

In the case of a linear associator with a quadratic error function, each input pattern and its associated target output define a hyperplane in weight space. If all hyperplanes intersect at a common point, then there is a unique exact solution to the association problem. If the hyperplanes do not intersect at a common point, it is still possible to look for that point in weight space which minimizes the quadratic error using on-line

backpropagation. The optimal solution can also be found by calculating the pseudoinverse of the matrix whose rows are the input patterns.
Figure 6 shows the iteration path of the learning algorithm for a two-dimensional example in which the target hyperplanes are the sides of a triangle. Each white point represents one iteration in weight space. It is easy to show that in this case on-line backpropagation behaves like a variant of the Gauß-Seidel iteration method and that at each iteration point an affine transformation is applied [5]. The set of affine transformations defined by the input patterns defines an *Iterated Function System* of the kind described by Barnsley [1]. This is sufficient proof that the iteration path in weight space is indeed a fractal, as Figure 6 shows for two different learning rates.

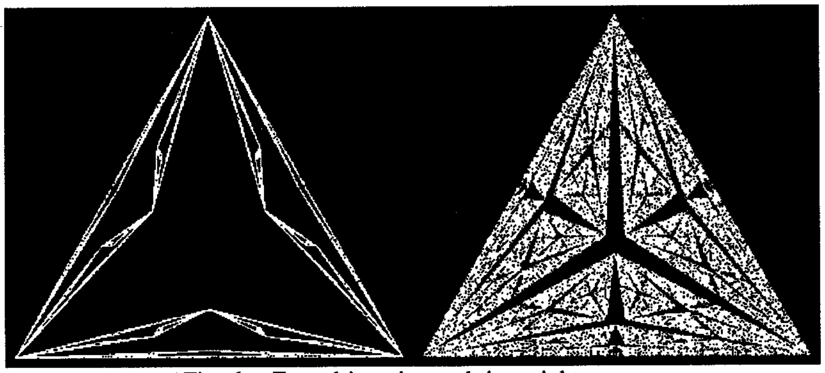
Fig. 6. Fractal iteration path in weight space

5. Backpropagation with momentum

In the case of off-line backpropagation with momentum, the iteration path in weight space is not so chaotic but also shows some kind of structure.
Figure 7 shows some examples of the iteration path in weight space for different combinations of learning and momentum rate. The more chaotic paths were produced by large momentum rates. Of course, these large momentum rates can be avoided, but in this case the range of values of the learning rate which lead to convergence decrease accordingly. Normally the optimal size of the learning rate must be adjusted by trial and error and depends on the relative magnitudes of the eigenvalues of the correlation matrix of the input patterns. Since these magnitudes are unknown and can differ dramatically, it is easy to miss the convergence region for the learning rate, in which case only a larger momentum rate can be of any help. It is instructive to know that by adjusting the relative magnitudes of learning and momentum rates, we are effectively choosing between one of the structured paths shown in Figure 7.

6. Conclusions

We have shown in this paper that there is more structure in the iteration path followed by some learning algorithms than normally thought. The visualization of the iteration path helps to understand the trade-offs associated with different combinations of learning parameters. In the case of perceptron learning, the boolean sphere helps to visualize perceptron learning and immediately suggests an optimization (bipolar vectors) which indeed works. It would be interesting to look at the case of non-linear activation functions and non-quadratic error functions and the iteration paths they produce in weight space. Some structure can also be found in this case, but this problem will be handled elsewhere.

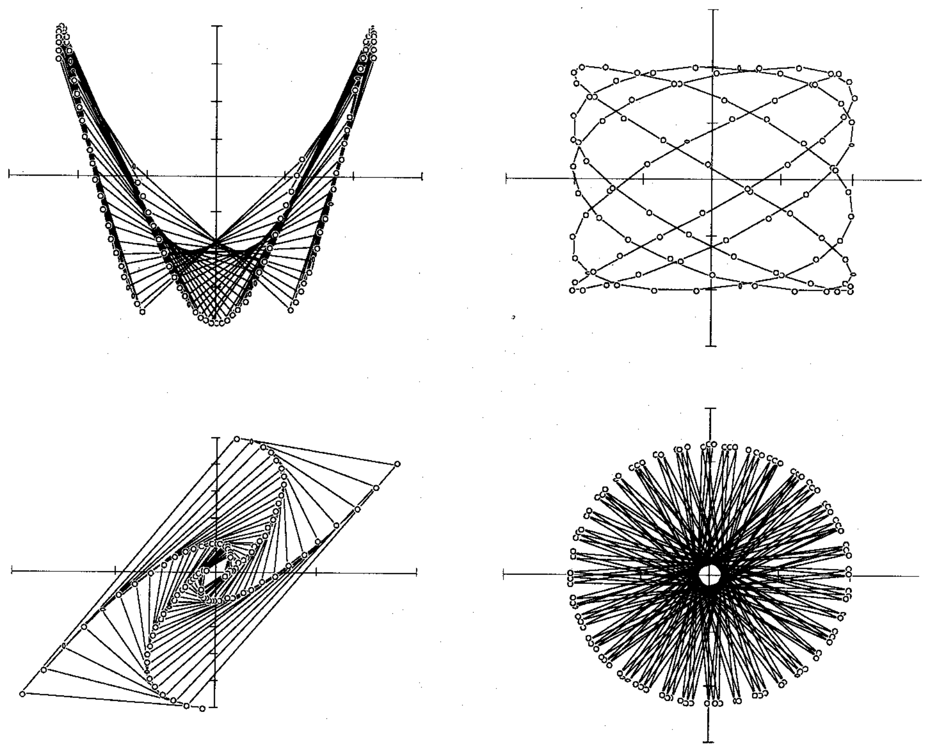

Fig. 7. Iteration paths for off-line backpropagation with momentum

References

[1] M. Barnsley, *Fractals Everywhere*, Academic Press, London, 1988.
[2] D. R. Hush, H. M. Salas and B. Horne, "Error Surfaces for Multilayer Perceptrons", *IEEE IJCNN*, Seattle, 1991, pp. I-759-761.
[3] H. van der Maas, P. Verschure, P. Molenaar, "A Note on Chaotic Behavior in Simple Neural Networks", Neural Networks, Vol. 3, 1990, pp. 119-122.
[4] R. Rojas, *Theorie der neuronalen Netze*, Springer-Verlag, 1993.
[5] R. Rojas, "The fractal geometry of backpropagation", submitted to the *IEEE International Joint Conference on Neural Networks 1994.*