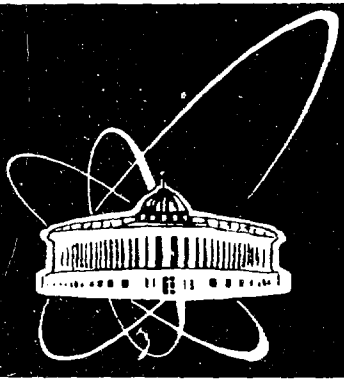

объединенный
институт
ядерных
исследований
дубна

D11-93-317

V.V.Ivanov, I.V.Puzynin, B.Purevdorj

A NEWTON-TYPE NEURAL NETWORK LEARNING ALGORITHM

Submitted to «Computer Physics Communications»

1993

Introduction

Today artificial neural networks are widely used in high energy physics [1, 2]. They are applied to both constructing high-level triggers providing effective discrimination of background events in the real time of experiment and processing collected data, e.g. track recognition, mass reconstruction, primary and/or secondary vertex identification.

In particular, feed-forward multilayer [4, 5] networks are widespread. Usually network learning is implemented with the back-propagation algorithm [3, 6] based on the gradient descent. However, this algorithm does not ensure high recognition accuracy (see discussion below). It is important for problems in experimental particle physics, where rare events are separated from suppressing background.

From the mathematical viewpoint the network learning consists in functional minimization [3]. In this paper a Newton-type learning algorithm is proposed and compared with first-order methods.

1 Problem statement

1.1 Network architecture

Feed-forward network involves an input layer corresponding to analyzed data, an output layer corresponding to results and hidden ones.

The output signal $\vec{y}$ is given by

$$y_i = f(a_i), \qquad a_i = \sum_j \omega_{ij} h_j + \theta_i,$$

$$h_j = \begin{cases} f(a_j) & \text{for hidden layers,} \\ x_j & \text{for input layer,} \end{cases}$$

$$f(x) = g(x/T),$$

where x_j - input data, ω_{ij} - neuron connection weights, θ_i - thresholds, T - temperature, g - transfer function. We choose

$$g(x) = \frac{1}{2}(1 + tanh(x)).$$

The learning procedure consists in finding weights minimizing the energy functional

$$E = \frac{1}{2} \sum_p \left(\vec{y}^{(p)} - \vec{t}^{(p)}\right)^2,$$

where $p = 1, \ldots, N_{train}$, N_{train} - number of training patterns, $\vec{t}$ - target value of output signal.

The network is simulated on IBM PC AT386/486 in the NDP environment [7], using JETNET 2.0 package developed in the Lund University [6].

1.2 Classification problem

We use neural networks for classifying samples from two overlapping distributions with densities f_1 and f_2. A network with the number of input neurons equal to the dimension of the events and one output neuron is taken. The learning set contains an equal quantity of events of both kinds. The target value is set to 0 for the 1st distribution and 1 for the 2nd one.

Denoting $y = F(\omega, x)$ one has in continuous limit

$$E = \int \left((F(\omega,x))^2 f_1(x) + (1 - F(\omega,x))^2 f_2(x) \right) dx.$$

Let us take

$$opt(x) = \frac{f_2(x)}{f_1(x) + f_2(x)}, \qquad D = \{x | f(x) = f_1(x) + f_2(x) \neq 0\}.$$

Then

$$E(opt + \epsilon) = E(opt) + \int \epsilon(x)^2 f(x) dx.$$

Introducing in D the norm $y = \left(\int y^2(x) f(x) dx \right)^{\frac{1}{2}}$, we see that minimization of E is equivalent to finding the function y nearest to opt. We consider accuracy of minimization as a learning quality.

In our numeric experiments multidimensional Gaussians are used as f_1 and f_2.

2 First-order methods

In the back-propagation algorithm (BP) described in [3] weights are calculated according to

1. Initial values are chosen randomly in the range $[-\omega_0, \omega_0]$.

2. Weight updates are given by

$$\Delta\vec{\omega}_n = -\eta \nabla E_n + \alpha \Delta\vec{\omega}_{n-1},$$

$$\frac{\partial E}{\partial \omega_{ij}} = d_i h_j, \qquad d_i = \begin{cases} (y_i - t_i) f'(a_i) & \text{for output layer,} \\ \sum_k d_k \omega_{ki} f'(a_i) & \text{for hidden layers.} \end{cases}$$

BP differs from the pure gradient descent in ∇E_n being calculated on a subset of the training set containing N patterns and a momentum α, important for ravine functions [3].

Efficiency of BP depends on the network parameters. However, it is difficult to formulate accurate criteria for choosing them. Some qualitative remarks are given below.

Optimal size of the network can be found proceeding from the geometry of the problem. The more complex regions to be bound with the network [4] are, the larger

the needed size is. Many problems[1] require small networks and increase of the size does not improve the learning quality.

The parameter N must satisfy the condition $N \ll N_{train}$. Learning efficiency increases if N patterns for the next step are chosen randomly. The parameter ω_0 should not be taken close to zero, since $E(\vec{\omega})$ is more ravine near 0, and far from zero, since the gradient norm is most probably very small. The stepleugth η is usually chosen empirically. η may be decreased for larger networks and learning sets. Near the solution it is useful to decrease η and increase T to lessen oscillations of the output signal during learning. The method for updating is not important.

It should be noted that usually the function E has no strongly marked local minima, which ensures the convergence of the method to the solution.

BP comes to the gradient descent when $N = N_{train}$ and $\alpha = 0$. In this case learning is very slow. η must be small and attempt to find it minimizing $E(\eta)$ at $0 < \theta \leq \eta \leq 1$ with E04JAF or E04ABF subroutines from NAG Library [8] does not affect the result strongly. The gradient descent can be expedient near the solution due to less oscillations.

Though BP is rather efficient, it has the following shortcomings:

- the method does not provide accurate approximation of the minimum;
- the learning parameters are often to be chosen empirically;
- it is difficult to detect reaching the neighborhood of the minimum;
- the gradient norm reduces slowly and is substantially non-zero leading to large oscillations.

3 Second-order methods

3.1 Newton method

In the Newton method weights are updated according to

$$\Delta\omega = -\eta(\nabla^2 E)^{-1}\nabla E,$$

where the Hessian matrix $\nabla^2 E$ is given by

$$\frac{\partial^2 E}{\partial\omega_{ij}\partial\omega_{mn}} = d_i\frac{\partial h_j}{\partial\omega_{mn}} + \frac{\partial d_i}{\partial\omega_{mn}}h_j.$$

The method is not applicable for this problem since $\nabla^2 E$ is signvariable and nearly singular in the neighborhood of the minimum.

We point out one of the possibilities of degeneration when the classified distributions are radially symmetrical ones with the same centre.

[1] For instance, most of the multidimensional data analysis problems in high energy physics.

Let us note that $F(\vec{\omega},\vec{x}) = F(\vec{\omega}_1\cdot\vec{x},\ldots,\vec{\omega}_n\cdot\vec{x},\vec{\omega}_0)$, where $\vec{\omega}_i$ - subvector containi weights connecting the input neurons with the i-th hidden one, $\vec{\omega}_0$ - remaining weigh Acting on $\vec{\omega}_i$ with the rotation matrix U, one has in continuous limit

$$E(\vec{\omega}') = \int F(\vec{\omega}',\vec{x}')d\vec{x}' = \int F(U\vec{\omega}_1,\ldots,U\vec{\omega}_n,\vec{\omega}_0,U\vec{x})d\vec{x} = \int F(\vec{\omega},\vec{x})d\vec{x} = E(\vec{\omega}).$$

When the rotation angle approaches zero, one has $\vec{\omega}_n \to \vec{\omega}$, $E(\vec{\omega}_n) = E(\vec{\omega})$.

Strictly speaking, the minimum is not completely degenerate, but the Hessian m trix is ill-conditioned near the minimum. Typical conditionality values $|\lambda|_{max}/|\lambda|_{min}$ - eigenvalues) are 10^3 at the initial point and $10^5 - 10^6$ (even 10^{12}) in the neighborho of the minimum.

The use of regularization [10]

$$\Delta\omega = -\eta H^{-1}A^*\nabla E,$$

$$H = A^*A + \alpha\|\nabla E\|^2, \qquad A = \nabla^2 E.$$

ensures convergence only with a large α ($\alpha \sim 1$) and does not provide better quali than a gradient descent. The matrix H is too ill-conditioned when the regularizi addition reduces to zero. Therefore, H cannot be inversed with acceptable accuracy

3.2 Newton method with Hessian correction

When the Hessian matrix has poor qualities the following method is used

$$\Delta\omega = -\eta H^{-1}\nabla E,$$

where H is positive-definite and well-conditioned. When $\|(\nabla^2 E)^{-1}\| > A$ in the sm neighborhood of the minimum, where A - a maximal acceptable value, $H \not\to \nabla^2 E$ wi approaching the minimum.

The matrix H is taken as [9]

$$H = \nabla^2 E + \mu I,$$

where μ ensures that eigenvalues of $H \geq \delta > 0$. The choice $\delta = 1$ gives good resul In this condition H is a squeezing operator.

The steplength η is calculated by means of one-dimension optimization (see Se tion 2). Let us note that the CPU time is spent mostly on the calculation of the Hessi matrix. Stabilization of η near 1 indicates, as a rule, that the solution is close. It also convenient to calculate η as [11]

$$\eta_{n+1} = \eta_n \frac{\|\nabla E_{n-1}\|}{\|\nabla E_n\|}, \quad 0 \leq \eta_{n+1} \leq 1,$$

where η_0 must be sufficiently small. However, it slows down the performance.

The method has the following merits:

- it provides the learning quality unachievable with BP. It is demonstrated by the value of E and the gradient norm, which is much closer to zero;
- as a rule, the convergence speed does not depend strongly on the networksize, the number of training patterns or complexity of the recognition problem;
- the solution of more complex problems, e.g. separation of strongly overlapped distributions or reliable recognition with a small learning set, is possible;
- the learning parameters are chosen automatically.

Large computational expenses are a demerit of the method. BP requires memory proportional to n and the CPU time per epoch – to $n \times N_{train}$, where n – the number of weights. The Newton method requires memory proportional to n^2 and the CPU time to $n^2 \times N_{train}$ (when $n \ll N_{train}$). Let us note that the learning speed is not a crucial factor if the network is used many times.

4 Comparison of the learning algorithms

As mentioned above, the feed-forward network is used for classifying 2 distributions. We present the performance of different methods in the table. The network configuration is 2-6-1. The following parameters are chosen: $\omega_0 = 0.5$, $T = 1$; and for BP: $\eta_0 = 0.1$, $\alpha = 0.5$, $N = 10$.

Table: Performance of different algorithms of the feed-forward network learning

Method	*Example*	N_{train}	*Limit*	*Epoch*	E_{min}	$\|\nabla E\|$
BP	1/0.3	1000	76,1	7	0.160*	1.0 – 20
NW				7	0.155*	0.047
GRAD				107	0.161*	0.48
REG				–	0.159*	0.0014
BP	1/0.7	1000	58,5	52	0.235	2.0 – 40
NW				4	0.229	0.13
BP	1/0.8	1000	55,3	–	0.242	4.0 – 40
NW				6	0.238	0.092
BP	1/0.3	100	76,1	19	0.144	0.4 – 4.0
NW				17	0.138	0.065
BP	1/0.5	100	66,1	–	0.182	0.2 – 5.0
NW				25	0.142	0.26
BP	1/0.3	5000	76,1	4	0.164	3.0 – 60
NW				7	0.163	0.10

The denotations are: BP – back-propagation, GRAD – pure gradient descent, NW – Newton method with Hessian correction, REG – Newton method with regularization

(used after 10 epochs of BP), *Example* – standard deviations of classified 2-dimension Gaussians (mean values are set to zero), *Limit* – Bayesian limit of recognition, *Epoch* – the number of epochs needed to reach the Bayesian limit and become stable, E_{min} – the least value of E reached during 100 (* – 1000) epochs and divided by N_{train}, $\|\nabla E\|$ – the gradient norm at the end of learning.

The figure shows learning curves for BP and the Newton-type method.

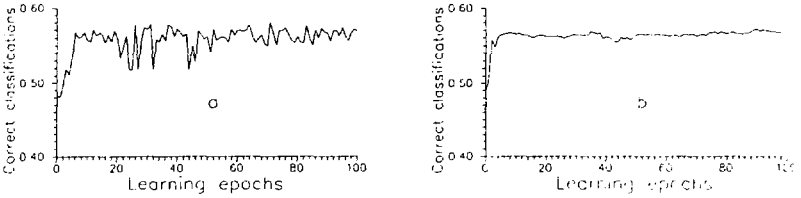

Fig. Fraction of correct classifications as a function of learning epochs for $Example = 1/0.7$ and $N_{train} = 1000$: a) BP, b) Newton-type method

Conclusion

The results of comparison of different learning algorithms are as follows.

1. When the learning parameters are optimal, BP is rather fast and requires a small memory size.
2. The pure gradient descent is expedient only near the solution.
3. The pure Newton method is not applicable.
4. The Newton method with the Hessian correction requires more computer resources than BP, but provides better learning quality and is simpler in use.

BP is better for big networks used for recognition of complex patterns, but the Newton-type method is good for relatively small networks when maximal recognition accuracy is needed.

References

[1] B.Denby: *"Tutorial on Neural Networks Applications in High Energy Physics: 1982 Perspective"*. In Proc. of the Second International Workshop on *"Software Engineering, Artificial Intelligence and Expert Systems in High Energy Physics"*.

January 13-18, 1992 L'Agelaude France-Telecom La Londe-les-Maures (France). New Computing Techniques in Physics Research II, edited by D.Perret-Gallix, World Scientific, 1992, p.287.

[2] C.Peterson: *"Neural Networks in High Energy Physics - Algorithms and Results"*, ibidem, p.327.

[3] D.E.Rumelhart, G.E.Hinton, R.J.Williams: *"Learning Internal Representations by Error Propagation"* in D.E.Rumelhart & J.L.McClelland (Eds.), *Parrallel Distributed Processing: Explorations in the Microstructure of Cognition.* Vol. 1: *Foundations.* MIT Press, 1986.

[4] R.P.Lippmann: *"An Introduction to Computing with Neural Nets"*, IEEE ASSP Mag.4, 1987, p.4.

[5] B.Humpert: *"A Comparative Study of Neural Network Architectures"*, Comp. Phys. Comm. 58, 1990, p.223.

[6] L.Lönnblad, C.Petterson, T.Rögnvaldsson: *"Pattern Recognition in High Energy Physics with Artificial Neural Networks - JETNET 2.0"*, Comp. Phys. Comm. 70, 1992, p.167.

[7] NDP - FORTRAN Reference Manual. Microway Inc., Kingston, Massachusetts.

[8] NAG Fortran Library Manual. NAG Ltd., Oxford, 1990.

[9] M.Minoux: *Programmation Mathématique.* Bordas et C.N.E.T. - E.N.S.T., Paris, 1989.

[10] V.V.Ermakov, N.N.Kalitkin: *"An Optimal Step and Regularization of the Newton Method"* (in Russian), Zhur. Vych. Mat. i Mat. Fiz., Vol.21, 2, 1981, p.491.

[11] T.Janlav, I.V.Puzynin: *"On the Convergence of Iterations Based on the Continuous Analog of the Newton Method"* (in Russian), Zhur. Vych. Mat. i Mat. Fiz., Vol.32, 6, 1992, p.846.

Received by Publishing Department
on August 8, 1993.

SUBJECT CATEGORIES
OF THE JINR PUBLICATIONS

Index	Subject
1.	High energy experimental physics
2.	High energy theoretical physics
3.	Low energy experimental physics
4.	Low energy theoretical physics
5.	Mathematics
6.	Nuclear spectroscopy and radiochemistry
7.	Heavy ion physics
8.	Cryogenics
9.	Accelerators
10.	Automatization of data processing
11.	Computing mathematics and technique
12.	Chemistry
13.	Experimental techniques and methods
14.	Solid state physics. Liquids
15.	Experimental physics of nuclear reactions at low energies
16.	Health physics. Shieldings
17.	Theory of condensed matter
18.	Applied researches
19.	Biophysics

Иванов В.В., Пузынин И.В., Пурэвдорж Б. Д11-93-317

Алгоритм обучения нейронной сети на основе метода Ньютона

Рассмотрены методы 1-го и 2-го порядка для обучения многослойной нейронной сети «feed-forward» типа. Предложен алгоритм на основе метода Ньютона и проведено его сравнение с обычным методом «back-propagation». Показано, что предложенный алгоритм обеспечивает более высокое качество обучения. Даны рекомендации по использованию рассмотренных методов.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Препринт Объединенного института ядерных исследований. Дубна, 1993

Ivanov V.V., Puzynin I.V., Purevdorj B. D11-93-317

A Newton-Type Neural Network Learning Algorithm

First- and second-order learning methods for feed-forward multilayer networks are considered. A Newton-type algorithm is proposed and compared with the common back-propagation algorithm. It is shown that the proposed algorithm provides better learning quality. Some recommendations for their usage are given.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Preprint of the Joint Institute for Nuclear Research. Dubna, 1993

7 р.

Редактор Е.И.Хижняк. Макет Р.Д.Фоминой

Подписано в печать 17.09.93
Формат 60x90/16. Офсетная печать. Уч.-изд. листов 0,53
Тираж 130. Заказ 46668

Издательский отдел Объединенного института ядерных исследований
Дубна Московской области