2—P—1 Fast Back-Propagation Learning Methods for Neural Networks in Speech*

P.Haffner, A.Waibel and K.Shikano
(ATR Interpreting Telephony Research Laboratories)

1) Introduction

Several improvements to the back-propagation learning algorithm are proposed to achieve fast optimization of speech tasks. A suitable sigmoid function is chosen, the learning step is dynamically adjusted and the weights are updated more frequently. Speedups up to several order of magnitude are obtained. Training for the speaker-dependent recognition of the phonemes "B", "D" and "G" takes only a few minutes on an Alliant parallel computer, as opposed to the 4 days reported earlier[2], and yields a recognition rate of more than 98.6% correct.

2) Improving learning speed.

The problem is: given sample data, we define an energy function as the mean square difference between network output and desired output and we try to reduce this energy to zero within the smallest learning time. As a function of the synaptic weights, this energy function defines a complex surface and learning may be seen as a trajectory on this surface, moving down along the steepest slope, preferably toward a global minimum. There are several ways to easy convergence: model an Energy surface without flat spots with an appropriate choice on the sigmoid function; choose the maximum learning step size while controlling overshooting; increase the weight updating frequency.

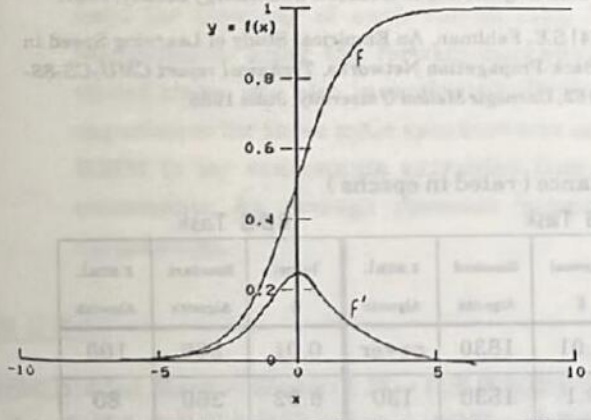

Fig 1 : Sigmoid function $f(x) = 1/(1+e^{-x})$ and its derivative.

2.1) The sigmoid function.

Back-Propagation Learning rate is proportional to the values of the sigmoid function f and its derivative f'. As seen in Fig.1, these functions flatten out at infinity, leading to very slow learning rate. A simple way to prevent this is to add some constants to f or f'.

2.2) Scaling the step size.

The optimal value of the step size may vary widely with time, which is consistent with the large variations of slope and curvature on the energy surface. As gauge of these variations, Franzini [3] has proposed the cosine of the angle between the error gradient at epoch t and that at epoch t-1: $\theta = \mathrm{Angle}(\nabla E(t-1), \nabla E(t))$. When learning with a momentum, $\theta = \mathrm{Angle}(\Delta w(t-1), -\nabla E(t))$ prevents much oscillations, even though Epsilon adaptation may be slower (Δw is the vector representing weight variation).

If we compute the angle over the set of input connections to unit u, the algorithm becomes local to this unit, updating the local step size ε_u according to $\varepsilon_u(t) = \varepsilon_u(t-1)e^{(\rho \cdot \cos(\theta))}$. In our task where units may have very different roles, we have found a large increase in learning speed using this method. ε may vary by a factor of 100 from one unit to another, depending mostly on the layer. This is probably due to the fact that learning dynamics vary widely from one layer to another.

While this algorithm does lead to significant improvements in speed, it is very sensitive to overshooting. During learning, very abrupt changes in learning strategy may be observed. We have added a control that, at each updating iteration, limits the norm of the vector $\varepsilon_u \nabla E$ to a fixed value $o = 1.0$.

2.3) Increase weight updating frequency.

Splitting a large and often highly redundant training set into smaller subsets for the purpose of weight updating may be very advantageous, assuming that these subsets are representative of the problem.

- At the beginning of the learning phase, one subset is enough data for a network which is only acting as a rough classifier. As a

*TDNNにおけるバックプロパゲーションアルゴリズムの高速化,
パトリック ハフナー、アレックス ワイベル、鹿野清宏 (ATR自動翻訳電話研究所)

consequence, updating weights over any subset may be as effective as updating weights over the whole training set.

- At the end of the learning phase, detailed learning and a large training set are needed. However, the difference between two learning subsets may introduce noise and help avoid local minima.

3) Experiments

An "epoch" corresponds to one presentation of the whole training set. As the size of our training set is fixed and as the algorithms we present do not modify substantially the computing time per epoch, we will use epoch to rate performance.

We have first dealt with mere learning speed of a training set, in which weights are updated at each epoch and the momentum is set to 0.9. Three tasks are evaluated: XOR: learning exclusive or in a neural network; 838: the network has to learn to encode 8 inputs within the 3 hidden units; BDG: learning the three stop consonants 'B','D' and 'G' in a TDNN (Time Delay Neural Network) from 250 training samples.

Results are shown in Table 1:

- With overshooting control, our Epsilon scaling algorithm improves the learning speed of all the tasks, particularly if the initial value of ε is set too small.

- Subtracting 0.5 to the sigmoid makes it symmetric and *generally* improves convergence.

- Adding a small positive value to the sigmoid derivative is useful to speed up slow convergence [4].

We ran these learning algorithms on larger training sets and found that generalization capacity on test data (different from training data) was a decreasing function of learning speed, particularly if we are using the heightened sigmoid derivative. This is closely related to the fact that both weights and weight variations become too large. We made these variations smaller by updating weights more frequently (here each 12 or 24 pattern presentation). Our network learned BDG task from 783 training samples and achieved a 98.6% recognition rate on test data whithin 20 epochs (less than 5 minutes on an Alliant computer, as opposed to the 4 days reported earlier for the same task[2]).

4) Conclusion

We have shown that learning time could be short for tasks like phoneme recognition in Neural Networks. These results are encouraging for the scope of tasks that can be handled by back-propagation in speech recognition. We have proposed some procedures for tuning learning parameters as the step size, the weight updating frequency and the shape of the sigmoid function to optimize the trade-off between learning speed and generalization capacity.

Acknowledgement

Authors would like to express their gratitude to Dr. Akira Kurematsu, president of ATR Interpreting Telephony Research Laboratories, for his encouragement and support. We are also indebted to the members of the Speech Processing Department.

References.

[1] D.E. Rumelhart and J.L. McClelland. *Parallel distributed Processing; Explorations in the Microstructures of Cognition.* Volume I and II, MIT press, Cambridge, MA, 1986.

[2] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano and K. Lang. Phoneme Recognition Using Time-Delay Neural Networks. *Technical Report TR-1-0006, ATR Interpreting Telephony Research Laboratories, October 1987.*

[3] M.A. Franzini. Speech recognition with Back Propagation. In *Proceedings, Ninth Annual Conference of IEEE Engineering in Medecine and Biology Society. 1987.*

[4] S.E. Fahlman. An Empirical Study of Learning Speed in Back-Propagation Networks. *Technical report CMU-CS-88-162, Carnegie Mellon University,* June 1988.

Table 1. Recognition Performance (rated in epochs)

		XOR Task			838 Task				BDG Task		
Sigmoid shift	Derivative add	Initial ε	Standard Algorith	ε scal. Algorith	Initial ε	Minimal ε	Standard Algorith	ε scal. Algorith	Initial ε	Standard Algorith	ε scal. Algorith
0	0	0.5	1010	70	0.1	0.01	1830	never	0.01	560	100
0	0	10	30	35	0.1	0.1	1830	130	0.02	360	80
-0.5	0	10	25	25	0.1	0.1	1525	130	0.02	315	80
-0.5	+0.1	10	215	35	0.1	0.1	640	80	0.02	190	80